

HiRDB Datareplicator Version 8
Description, User's Guide and Operator's
Guide

3020-6-360-50(E)

■ Relevant program products

List of program products

For the Windows Server 2003 x64 Editions, Windows Server 2008 R2, Windows Server 2008 (x64), Windows XP x64 Edition, Windows Vista Ultimate (x64), Windows Vista Business (x64), Windows Vista Enterprise (x64), Windows 7 Professional (x64), Windows 7 Enterprise (x64), or Windows 7 Ultimate (x64) operating system:

P-2462-1K87 HiRDB Datareplicator Version 8 08-06

For the Windows Server 2003 (IPF), or Windows Server 2008 (IPF) operating system:

P-2862-1K87 HiRDB Datareplicator Version 8(64) 08-06

■ Trademarks

AIX is a trademark of International Business Machines Corporation in the United States, other countries, or both.

AIX 5L is a trademark of International Business Machines Corporation in the United States, other countries, or both.

AMD, AMD Opteron, and combinations thereof, are trademarks of Advanced Micro Devices, Inc.

HACMP is a trademark of International Business Machines Corporation in the United States, other countries, or both.

HP-UX is a product name of Hewlett-Packard Development Company, L.P. in the U.S. and other countries.

Itanium is a trademark of Intel Corporation in the United States and other countries.

Linux^(R) is the registered trademark of Linus Torvalds in the U.S. and other countries.

Microsoft and SQL Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Microsoft and Visual Studio are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

ODBC is Microsoft's strategic interface for accessing databases.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

Red Hat is a trademark or a registered trademark of Red Hat Inc. in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Visual C++ is either a registered trademark or a trademark of Microsoft Corporation in the United States and/or other countries.

Windows is either a registered trademark or a trademark of Microsoft Corporation in the United States and/or other countries.

Windows Server is either a registered trademark or a trademark of Microsoft Corporation in the United States and/or other countries.

Windows Vista is either a registered trademark or a trademark of Microsoft Corporation in the United States and/or other countries.

Other product and company names mentioned in this document may be the trademarks of their respective owners. Throughout this document Hitachi has attempted to distinguish trademarks from descriptive terms by writing the name with the capitalization used by the manufacturer, or by writing the name with initial capital letters. Hitachi cannot attest to the accuracy of this information. Use of a trademark in this document should not be regarded as affecting the validity of the trademark.

■ Restrictions

Information in this document is subject to change without notice and does not represent a commitment on the part of Hitachi. The software described in this manual is furnished according to a license agreement with Hitachi. The license agreement contains all of the terms and conditions governing your use of the software and documentation, including all warranty rights, limitations of liability, and disclaimers of warranty.

Material contained in this document may describe Hitachi products or features not available in your country.

No part of this material may be reproduced in any form or by any means without permission in writing from the publisher.

■ Issued

Sep. 2013: 3020-6-360-50(E)

■ Copyright

All rights reserved. Copyright (C) 2007, 2013, Hitachi, Ltd.

Preface

This manual explains how to use the following program products:

- P-2462-1K87 HiRDB Datareplicator Version 8
- P-2862-1K87 HiRDB Datareplicator Version 8

This program product is referred to generically in this manual as Datareplicator.

Note that the English version of HiRDB Datareplicator is able to perform data linkage only between HiRDB databases. It cannot perform data linkage with the following databases:

- XDM/RD E2
- TMS-4V/SP
- RDB I E2
- XDM/SD E2
- ADM
- PDMII E2

Additionally, the English version of HiRDB Datareplicator Version 8 is supported only in the Windows edition of HiRDB. Therefore, although this manual includes explanations of the UNIX edition of HiRDB Datareplicator, no English version is supported in UNIX.

Datareplicator can be installed in a HiRDB (HiRDB/Single Server or HiRDB/Parallel Server) Version 9 or later.

Intended readers

This manual is intended for system administrators, system designers, programmers, and operators who use Datareplicator to construct or operate a data linkage system.

The manual assumes that you are a user who has the following knowledge:

User who constructs/operates a HiRDB-to-HiRDB data linkage system:

- Basic knowledge of the operating system under which Datareplicator runs
- Knowledge required of a HiRDB system administrator

Organization of the manual

This manual is organized as follows:

1. Overview

Chapter 1 describes the features of Datareplicator, data linkage, the software configuration, and the procedure for constructing a data linkage system.

2. Environment Setup

Chapter 2 describes the products associated with Datareplicator, the Datareplicator installation procedure and the directories that must be created, the specification of environment variables, and the communications environment.

3. Data Linkage Facilities

Chapter 3 describes the data linkage patterns supported by Datareplicator, the facilities for data linkage extraction and import processing, and the functions provided by Datareplicator.

4. System Design

Chapter 4 explains the procedures for designing a data linkage system appropriate to the application mode, as well as the procedures for designing source and target Datareplicators.

5. Definitions

Chapter 5 explains the procedures for defining the design of a data linkage system between source and target Datareplicators and provides definition examples.

6. Operation

Chapter 6 explains the procedures for operating the data linkage system, and explains how to start, terminate, and operate Datareplicator.

7. Command Syntax

Chapter 7 explains Datareplicator's command syntax.

8. User Own Coding Routines

Chapter 8 provides an overview of user own coding routines supported by Datareplicator and explains the creation and function syntax for user own coding routines; it also provides coding examples.

9. Error Handling Procedures

Chapter 9 explains the procedures for handling errors that occur during operation of source and target Datareplicators.

10. Messages

Chapter 10 explains the message output format and the handling of messages.

A. Detailed Information About HiRDB-Related Datareplicator Support

Appendix A lists the versions of Datareplicator that can be used, classified by the

operating systems that are supported for HiRDB. Appendix A also lists the support provided by Datareplicator for Datareplicator-related HiRDB facilities.

B. Datareplicator Reserved Words

Appendix B provides a list of the Datareplicator reserved words.

C. Functional Differences Between the UNIX and Windows Editions of Datareplicator

Appendix C describes the functional differences between the UNIX and Windows editions of Datareplicator.

D. Downgrading Datareplicator

Appendix D explains how to restore Datareplicator to a previous version.

E. Glossary

Appendix E is a glossary that explains the terms used in the Datareplicator and Datareplicator Extension manuals.

Related publications

This manual is part of a related set of manuals. The manuals in the set are listed below (with the manual numbers).

HiRDB (for UNIX)

- *HiRDB Version 9 Description* (3000-6-451(E))
- *HiRDB Version 9 Installation and Design Guide* (3000-6-452(E))
- *HiRDB Version 9 System Definition* (3000-6-453(E))
- *HiRDB Version 9 System Operation Guide* (3000-6-454(E))
- *HiRDB Version 9 Command Reference* (3000-6-455(E))
- *HiRDB Version 9 Staticizer Option Description and User's Guide* (3000-6-463), for UNIX systems[#]
- *HiRDB Version 9 Disaster Recovery System Configuration and Operation Guide* (3000-6-464(E)), for UNIX systems
- *HiRDB Version 9 Batch Job Accelerator* (3020-6-468), for UNIX systems[#]
- *HiRDB Version 9 Memory Database Installation and Operation Guide* (3020-6-469), for UNIX systems[#]

HiRDB (for Windows)

- *HiRDB Version 9 Description* (3020-6-451(E))
- *HiRDB Version 9 Installation and Design Guide* (3020-6-452(E))

- *HiRDB Version 9 System Definition* (3020-6-453)[#]
- *HiRDB Version 9 System Operation Guide* (3020-6-454(E))
- *HiRDB Version 9 Command Reference* (3020-6-455)[#]
- *HiRDB Version 8 Batch Job Accelerator* (3020-6-368)[#]

HiRDB (for UNIX and Windows)

- *HiRDB Version 9 UAP Development Guide* (3020-6-356(E))
- *HiRDB Version 9 SQL Reference* (3020-6-357(E))
- *HiRDB Version 9 Messages* (3020-6-358(E))
- *HiRDB Version 9 XDM/RD E2 Connection Facility* (3020-6-465)[#]
- *HiRDB Version 9 XML Extension* (3020-6-480)[#]
- *HiRDB Version 9 Text Search Plug-in* (3020-6-481)[#]
- *HiRDB Version 8 Security Guide* (3020-6-359)[#]
- *HiRDB Datareplicator Extension Version 8 Description, User's Guide and Operator's Guide* (3020-6-361)[#]
- *HiRDB Dataextractor Version 8 Description, User's Guide and Operator's Guide* (3020-6-362(E))

You must use the UNIX or the Windows manuals, as appropriate to the platform you are using.

Others

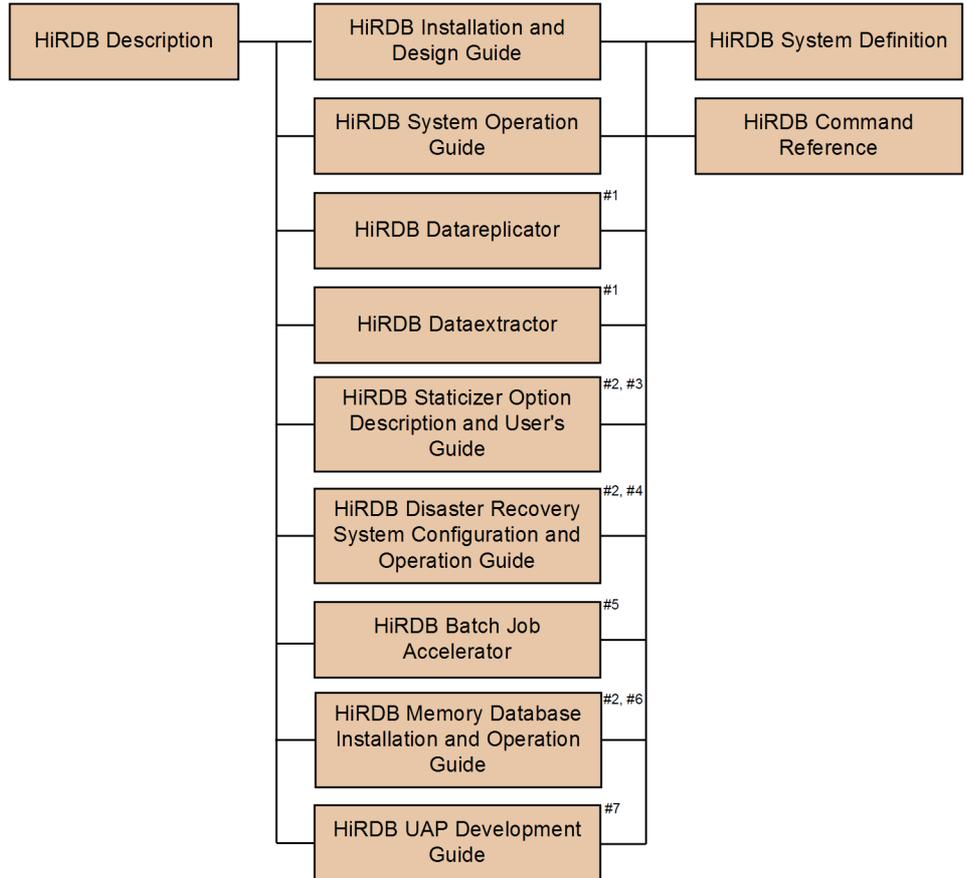
- *JP1 Version 9 JP1/Consolidated Management 2/Network Node Manager* (3020-3-L01)[#]
- *JP1 Version 9 JP1/Consolidated Management 2/Extensible SNMP Agent* (3020-3-L04)[#]
- *JP1 Version 9 JP1/Cm2/SNMP System Observer* (3020-3-L22)[#]

[#]: This manual has been published in Japanese only; it is not available in English.

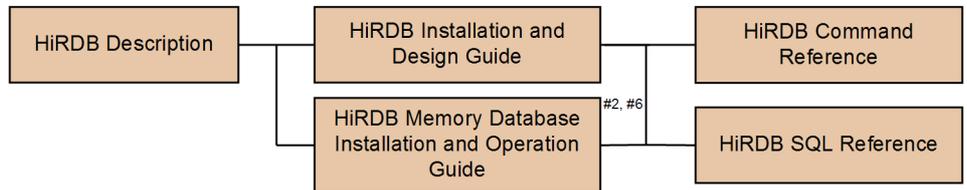
Organization of HiRDB manuals

The HiRDB manuals are organized as shown below. For the most efficient use of these manuals, it is suggested that they be read in the order they are shown, going from left to right.

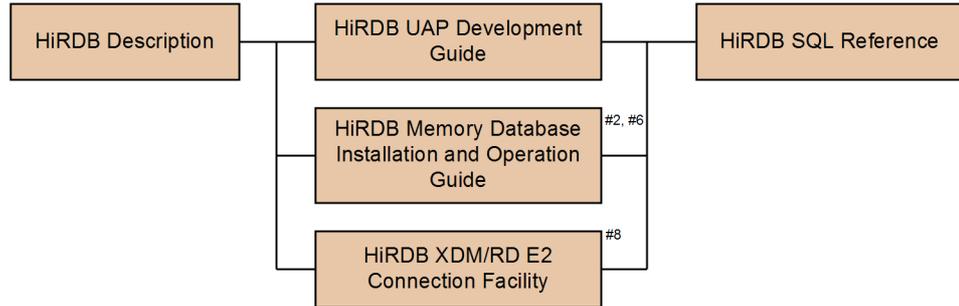
For system administrators:



For users who create tables:



For users who create or execute UAPs:



- #1: Read if you intend to use the replication facility to link data.
- #2: Published for UNIX only. There is no corresponding Windows manual.
- #3: Read if you intend to use the inner replica facility.
- #4: Read if you intend to configure a disaster recovery system.
- #5: Read if you intend to use in-memory data processing to accelerate batch operations.
- #6: Read if you intend to use memory database facility.
- #7: Must be read if you are linking HiRDB to an OLTP system.
- #8: Read if you intend to use the XDM/RD E2 connection facility to perform operations on XDM/RD E2 databases.

Conventions: Abbreviations

Unless otherwise required, this manual uses the following abbreviations for product and other names:

Full name or meaning	Abbreviation	
HiRDB Server Version 9	HiRDB/Single Server	HiRDB or HiRDB server
	HiRDB/Parallel Server	
HiRDB/Developer's Kit Version 9	HiRDB/Developer's Kit	HiRDB client
HiRDB/Developer's Kit Version 9 (64)		
HiRDB/Run Time Version 9	HiRDB/Run Time	
HiRDB/Run Time Version 9 (64)		
HiRDB Datareplicator Version 8, HiRDB Datareplicator Version 8 (64)	HiRDB Datareplicator	
HiRDB Dataextractor Version 8, HiRDB Dataextractor Version 8 (64)	HiRDB Dataextractor	
HiRDB Advanced High Availability Version 9	HiRDB Advanced High Availability	
HiRDB Accelerator Version 8	HiRDB Accelerator	

Full name or meaning	Abbreviation	
HiRDB Accelerator Version 9		
HiRDB Non Recover Front End Server Version 9	HiRDB Non Recover FES	
HiRDB Staticizer Option Version 9	HiRDB Staticizer Option	
HiRDB Text Search Plug-in Version 9	HiRDB Text Search Plug-in	
HiRDB XML Extension Version 9	HiRDB XML Extension	
Cm2/Extensible Agent, JP1/Cm2/Extensible SNMP Agent	JP1/Cm2	
HP-UX 11i V2 (IPF)	HP-UX or HP-UX (IPF)	
HP-UX 11i V3 (IPF)		
HP-UX 11i V2 (PA-RISC)	HP-UX	
AIX 5L V5.1	AIX 5L	AIX
AIX 5L V5.2		
AIX 5L V5.3		
AIX V6.1	AIX V6.1	
AIX V7.1	AIX V7.1	
Red Hat Linux	Red Hat Linux	
Red Hat Enterprise Linux	Red Hat Enterprise Linux	
Red Hat Enterprise Linux AS 2.1		
Red Hat Enterprise Linux AS 3 (x86)		
Red Hat Enterprise Linux ES 3 (x86)		
Red Hat Enterprise Linux ^(R) AS 3 (IPF)	Linux (IPF)	
Red Hat Enterprise Linux ^(R) AS 4 (IPF)	Linux (EM64T)	
Red Hat Enterprise Linux ^(R) AS 3 (AMD64 & Intel EM64T)		
Red Hat Enterprise Linux ^(R) AS 4 (AMD64 & Intel EM64T)		
Red Hat Enterprise Linux ^(R) ES 3 (AMD64 & Intel EM64T)		
Red Hat Enterprise Linux ^(R) ES 4 (AMD64 & Intel EM64T)		
Red Hat Enterprise Linux ^(R) 5.1 Advanced Platform (x86)	Linux 5.1	

Full name or meaning	Abbreviation			
Red Hat Enterprise Linux ^(R) 5.1 (x86)				
Red Hat Enterprise Linux ^(R) 5.1 Advanced Platform (AMD/Intel 64)				
Red Hat Enterprise Linux ^(R) 5.1 (AMD/Intel 64)				
Red Hat Enterprise Linux ^(R) 5.1 Advanced Platform (Intel Itanium)				
Red Hat Enterprise Linux ^(R) 5.1 (Intel Itanium)				
Red Hat Enterprise Linux 6 (32-bit x86)	Linux 6			
Red Hat Enterprise Linux 6 (64-bit x86_64)				
Red Hat Linux 7.1	Red Hat Linux 7.1			
Red Hat Linux 7.2	Red Hat Linux 7.2			
Microsoft ^(R) Windows ^(R) 2000 Professional Operating System	Windows 2000			
Microsoft ^(R) Windows ^(R) 2000 Server Operating System				
Microsoft ^(R) Windows ^(R) 2000 Advanced Server Operating System				
Microsoft ^(R) Windows ^(R) 2000 Datacenter Server Operating System				
Microsoft ^(R) Windows ^(R) XP Professional Operating System	Windows XP Professional	Windows XP		
Microsoft ^(R) Windows ^(R) XP Professional x64 Edition	Windows XP x64 Edition			
Microsoft ^(R) Windows Server ^(R) 2003, Standard Edition	Windows Server 2003 Standard Edition	Windows Server 2003		
Microsoft ^(R) Windows Server ^(R) 2003, Enterprise Edition	Windows Server 2003 Enterprise Edition			
Microsoft ^(R) Windows Server ^(R) 2003, Standard x64 Edition	Windows Server 2003 Standard x64 Edition			
Microsoft ^(R) Windows Server ^(R) 2003, Enterprise x64 Edition	Windows Server 2003 Enterprise x64 Edition			

Full name or meaning	Abbreviation	
Microsoft ^(R) Windows Server ^(R) 2003, Enterprise x64 Edition (IPF)	Windows Server 2003 (IPF)	
Microsoft ^(R) Windows Server ^(R) 2003 R2, Standard Edition	Windows Server 2003 R2	
Microsoft ^(R) Windows Server ^(R) 2003 R2, Enterprise Edition		
Microsoft ^(R) Windows Server ^(R) 2003 R2, Standard x64 Edition		
Microsoft ^(R) Windows Server ^(R) 2003 R2, Enterprise x64 Edition	Windows Server 2003 R2 x64 Editions	
Microsoft ^(R) Windows Server ^(R) 2003 R2, Standard x64 Edition		
Microsoft ^(R) Windows Server ^(R) 2003 R2, Enterprise x64 Edition		
Microsoft ^(R) Windows Vista ^(R) Business		Windows Vista
Microsoft ^(R) Windows Vista ^(R) Enterprise		
Microsoft ^(R) Windows Vista ^(R) Ultimate		
Microsoft ^(R) Windows Server ^(R) 2008 Standard	Windows Server 2008 Standard	Windows Server 2008
Microsoft ^(R) Windows Server ^(R) 2008 Enterprise	Windows Server 2008 Enterprise	
Microsoft ^(R) Windows Server ^(R) 2008, Enterprise x64 Edition (IPF)	Windows Server 2008 (IPF)	
Microsoft ^(R) Windows ^(R) 7 Home Premium	Windows 7	
Microsoft ^(R) Windows ^(R) 7 Professional		
Microsoft ^(R) Windows ^(R) 7 Enterprise		
Microsoft ^(R) Windows ^(R) 7 Ultimate		
Microsoft ^(R) Windows ^(R) 7 Home Premium (x64)		
Microsoft ^(R) Windows ^(R) 7 Professional (x64)		
Microsoft ^(R) Windows ^(R) 7 Enterprise (x64)		

Full name or meaning	Abbreviation
Microsoft ^(R) Windows ^(R) 7 Ultimate (x64)	
Microsoft ^(R) SQL Server	SQL Server
Oracle9i	
Oracle 10g	
Oracle Database 11g	
VOS3/US, VOS3/LS, VOS3/FS, VOS3/AS, VOS3/ES1, and the systems on which these OSs run	VOS3
VOS1/LS, VOS1/FS, VOS1/ES2, and the systems on which these OSs run	VOS1

- Windows 2000, Windows XP, Windows Server 2003, Windows Vista, Windows Server 2008, Windows 7, and Windows 7 (x64) are often referred to collectively as *Windows*. The individual operating systems are referred to explicitly only if required for purposes of clarity.
- JP1/Cm2 and NETM*Cm2 are often referred to collectively as *JP1/Cm2*. Explanations about JP1/Cm2 also apply to the corresponding NETM*Cm2 products.

This manual also uses the following abbreviations:

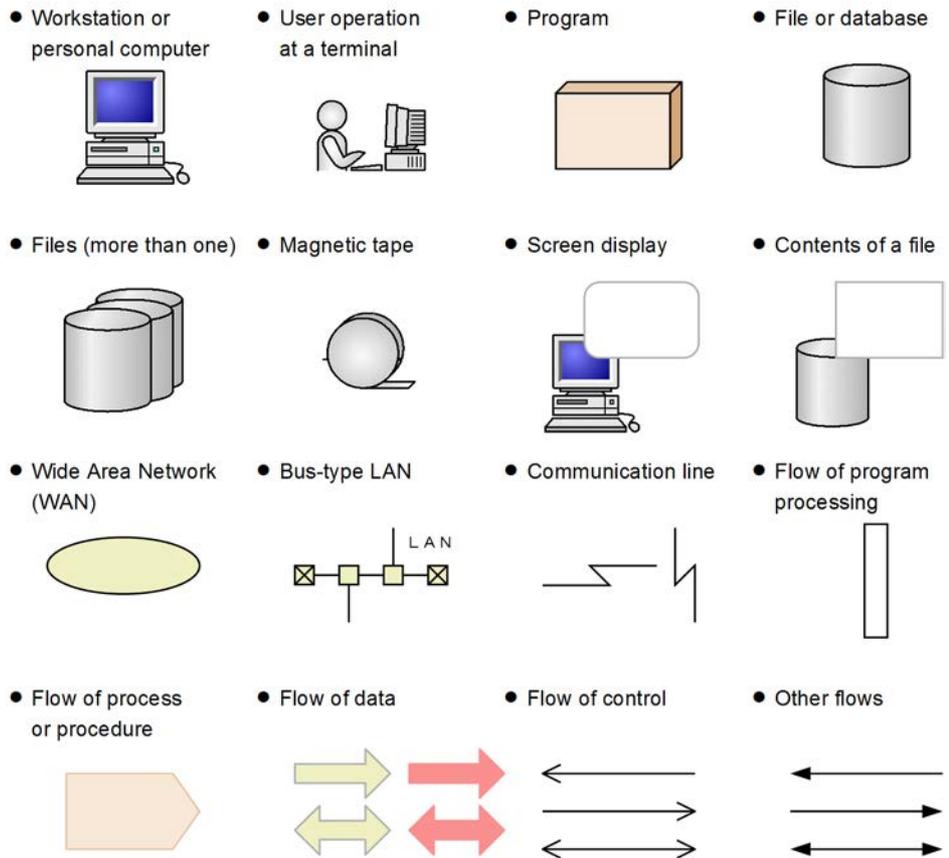
Abbreviation	Meaning
ADM	Adaptable Data Manager
ADT	Abstract Data Type
AP	Application Program
API	Application Programming Interface
BES	Back End Server
BLOB	Binary Large Object
CD-ROM	Compact Disc - Read Only Memory
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DAT	Digital Audio Tape recorder
DB	Database
DBM	Database Module

Abbreviation	Meaning
DNS	Domain Name Service
DS	Dictionary Server
DWH	Data Warehouse
EBCDIC	Extended Binary Coded Decimal Interchange Code
EBCDIK	Extended Binary Coded Decimal Interchange Kana code
EUC	Extended UNIX Code
FD	Floppy Disk
FES	Front End Server
HD	Hard Disk
HNA	Hitachi Network Architecture
IPF	Itanium ^(R) Processor Family
JFS	Journaled File System
JFS2	Enhanced Journaled File System
JIS	Japanese Industrial Standard code
JP1	Job Management Partner 1
KEIS	Kanji processing Extended Information System
LAN	Local Area Network
MGR	System Manager
MIB	Management Information Base
MSCS	Microsoft Cluster Server
OFIS/POL	Office Automation and Intelligence Support Software/Problem Oriented Language
OLAP	Online Analytical Processing
OS	Operating System
OSI	Open Systems Interconnection
PC	Personal Computer
PDM2 E2	Practical Data Manager 2 Extended Version 2
PP	Program Product

Abbreviation	Meaning
RDB1 E2	Relational Database Manager 1 Extended Version 2
SCSI	Small Computer System Interface
SDS	Single Database Server
SGML	Standard Generalized Markup Language
SNMP	Simple Network Management Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TMS-4V/SP	Transaction Management System - 4V/System Product
UAC	User Account Control
UAP	User Application Program
UOC	User Own Coding
VOS1	Virtual-storage Operating System 1
VOS3	Virtual-storage Operating System 3
WS	Workstation
XDM/BASEE2	Extensible Data Manager/Base Extended Version 2
XDM/DS	Extensible Data Manager/Data Spreader
XDM/RD E2	Extensible Data Manager/Relational Database Extended Version 2
XDM/SD E2	Extensible Data Manager/Structured Database Extended Version 2
XDM/XT	Extensible Data Manager/Data Extract
XML	Extensible Markup Language
XNF/S-E2	Extended HNA based communication Networking Facility/for Server - Extended Version 2

Conventions: Diagrams

This manual uses the following conventions in diagrams:



Conventions: Fonts and symbols

The following table explains the text formatting conventions used in this manual:

Text formatting	Convention
Bold	<p>Bold characters indicate text in a window, other than the window title. Such text includes menus, menu options, buttons, radio box options, or explanatory labels. For example:</p> <ul style="list-style-type: none"> • From the File menu, choose Open. • Click the Cancel button. • In the Enter name entry box, type your name.

Text formatting	Convention
<i>Italic</i>	<p>Italic characters indicate a placeholder for some actual text to be provided by the user or system. For example:</p> <ul style="list-style-type: none"> Write the command as follows: <code>copy source-file target-file</code> The following message appears: <code>A file was not found. (file = file-name)</code> <p>Italic characters are also used for emphasis. For example:</p> <ul style="list-style-type: none"> Do <i>not</i> delete the configuration file.
Monospace	<p>Monospace characters indicate text that the user enters without change, or text (such as messages) output by the system. For example:</p> <ul style="list-style-type: none"> At the prompt, enter <code>dir</code>. Use the <code>send</code> command to send mail. The following message is displayed: <code>The password is incorrect.</code>

The following table explains the symbols used in this manual:

Symbol	Convention
	<p>In syntax explanations, a vertical bar separates multiple items, and has the meaning of OR. For example: <code>A B C</code> means A, or B, or C.</p>
{ }	<p>In syntax explanations, curly brackets indicate that only one of the enclosed items is to be selected. For example: <code>{A B C}</code> means only one of A, or B, or C.</p>
[]	<p>In syntax explanations, square brackets indicate that the enclosed item or items are optional. For example: <code>[A]</code> means that you can specify A or nothing. <code>[B C]</code> means that you can specify B, or C, or nothing.</p>
...	<p>In coding, an ellipsis (...) indicates that one or more lines of coding have been omitted.</p> <p>In syntax explanations, an ellipsis indicates that the immediately preceding item can be repeated as many times as necessary. For example: <code>A, B, B, ...</code> means that, after you specify A, B, you can specify B as many times as necessary.</p>
()	<p>Parentheses indicate the range of items to which the vertical bar () or ellipsis (...) is applicable.</p>
~	<p>The user-specified value preceding the swung dash must be specified in accordance with the attributes following the swung dash.</p>
<>	<p>Angle brackets enclose the syntax element notation for a user-specified value.</p>

Symbol	Convention
<<>>	Double angle brackets enclose the default value assumed by the system when the specification is omitted.
(())	Double parentheses enclose the permitted range of values that can be specified.
↑ ↑	The resulting value is to be rounded up. <i>Example</i> The result of $\uparrow 34 \div 3 \uparrow$ is 12.
↓ ↓	The resulting value is to be rounded off. <i>Example</i> The result of $\downarrow 34 \div 3 \downarrow$ is 11.
MAX	Largest value is to be selected. <i>Example</i> The result of $\text{MAX}(3 \times 6, 4 + 7)$ is 18.
MIN	Smallest value is to be selected. <i>Example</i> The result of $\text{MIN}(3 \times 6, 4 + 7)$ is 11.

Conventions for permitted characters

In most cases, only the following characters are permitted as syntax elements (if other characters are permitted, the manual will state this explicitly):

Type	Definition
alphabetic	Alphabetic characters (A to Z, a to z) and _ (underline)
alphabetic symbol	Alphabetic characters (A to Z, a to z), #, @, \
alphanumeric	Alphabetic characters and numeric characters (0 to 9)
alphanumeric symbol	Alphabetic symbols and numeric characters (0 to 9)
unsigned integer	Numeric characters (0 to 9)
hexadecimal	Numeric characters (0 to 9) and A to F (or a to f)
identifier	Alphanumeric character (A to Z, a to z) string beginning with an alphabetic character
symbolic name	Alphanumeric symbol string beginning with an alphabetic symbol
character string	String consisting of any characters
pathname	String consisting of one or more symbolic names, forward slashes (/), backslashes (\), and periods (.)

Type	Definition
filename	Character string consisting of one or more alphabetic characters (A to Z, a to z), numeric characters (0 to 9), periods (.), underlines (_), hyphens (-), and at marks (@) (maximum 30 characters) Use single-byte characters only. In UNIX, alphabetic characters are case-sensitive. Specification of pathnames depends on the conventions of the operating system used to install Datareplicator.

Note:

Use single-byte characters only. Alphabetic characters are case-sensitive (that is, lowercase alphabetic characters are distinguished from uppercase alphabetic characters).

Conventions: KB, MB, GB, and TB

This manual uses the following conventions:

- 1 KB (kilobyte) is 1,024 bytes.
- 1 MB (megabyte) is 1,024² bytes.
- 1 GB (gigabyte) is 1,024³ bytes.
- 1 TB (terabyte) is 1,024⁴ bytes.

Conventions: Log representations

- Windows edition

The application log that is displayed by Windows Event Viewer is referred to as the *event log*. The following procedure is used to view the event log.

To view the event log:

1. Choose **Start, Programs, Administrative Tools (Common)**, and then **Event Viewer**.
2. Choose **Log**, and then **Application**.

The application log is displayed. Messages with **HiRDB Dataextractor** displayed in the **Source** column are messages issued by HiRDB Dataextractor.

- UNIX edition

The OS log is referred to generically as *syslogfile*. *syslogfile* is the log output destination specified in `/etc/syslog.conf`. Typically, the following files are specified as *syslogfile*.

OS	File
HP-UX	/var/adm/syslog/syslog.log
Solaris	/var/adm/messages or /var/log/syslog
AIX	/var/adm/ras/syslog
Linux	/var/log/messages

Conventions: Notations used in explanations of Windows operations

In this manual, the term *directory* includes both of the Windows terms *directory* and *folder*, and path names are delimited by forward slashes (/).

Conventions: Path name representations

- A forward slash (/) is used to represent the path name delimiter. If you are using the Windows edition of Datareplicator, replace the forward slashes used in this manual with backslashes (\). Path names that differ between the Windows and UNIX editions are written separately with their actual delimiters.
- The HiRDB directory path is represented as \$PDDIR. However, the actual path names for the Windows and UNIX editions are different. When both are written, %PDDIR% is used to represent the HiRDB directory path name in the Windows edition. An example follows:

UNIX edition: \$PDDIR/client/lib/

Windows edition: %PDDIR%\CLIENT\UTL\

Conventions: Version numbers

The version numbers of Hitachi program products are usually written as two sets of two digits each, separated by a hyphen. For example:

- Version 1.00 (or 1.0) is written as 01-00.
- Version 2.05 is written as 02-05.
- Version 2.50 (or 2.5) is written as 02-50.
- Version 12.25 is written as 12-25.

The version number might be shown on the spine of a manual as *Ver. 2.00*, but the same version number would be written in the program as *02-00*.

Important notes on this manual

The following facilities are explained, but they are not supported:

- Distributed database facility
- Server mode system switchover facility[#]
- User server hot standby[#]
- Rapid system switchover facility[#]
- Standby-less system switchover (1:1) facility
- Standby-less system switchover (effects distributed) facility
- HiRDB External Data Access facility
- Inner replica facility
- Updatable online reorganization
- Sun Java System Directory Server linkage facility
- Simple setup tool
- Extended syslog facility
- Rapid batch facility
- Memory database facility
- Linkage with JP1/NETM/Audit

[#]: Although these facilities are not supported in the Windows edition of HiRDB, they are supported in the UNIX edition.

The following products and option program products are explained, but they are not supported:

- HiRDB CM
- HiRDB Disaster Recovery Light Edition
- uCosminexus Grid Processing Server
- HiRDB Text Search Plug-in
- HiRDB XML Extension
- TP1/Server Base
- JP1/PFM-Agent Option for HiRDB
- JP1/VERITAS NetBackup Agent for HiRDB License
- HiRDB Dataextractor
- HiRDB Datareplicator
- XDM/RD

- HiRDB SQL Tuning Advisor
- COBOL2002

Contents

Preface	i
Intended readers	i
Organization of the manual	i
Related publications	iii
Organization of HiRDB manuals	iv
Conventions: Abbreviations	vi
Conventions: Diagrams	xii
Conventions: Fonts and symbols.....	xiii
Conventions: KB, MB, GB, and TB	xvi
Conventions: Log representations	xvi
Conventions: Notations used in explanations of Windows operations	xvii
Conventions: Path name representations.....	xvii
Conventions: Version numbers.....	xvii
Important notes on this manual	xvii
1. Overview	1
1.1 Features.....	2
1.1.1 Purpose of Datareplicator.....	2
1.1.2 Data linkage system	4
1.2 Data linkage.....	9
1.2.1 Combinations for a data linkage system	9
1.2.2 Mechanism of data linkage	9
1.2.3 Correspondence between source and target databases	16
1.2.4 Databases supported for data linkage.....	16
1.2.5 Correspondence of database terminology	19
1.3 Tables supported for data linkage	21
1.4 Data types for tables supported for data linkage	23
1.4.1 Data types supported by Datareplicator	23
1.4.2 Data linkage for a table using an abstract data type.....	24
1.4.3 Data linkage for a table containing repetition columns.....	26
1.4.4 Data linkage for tables using BLOB and BINARY type columns	27
1.5 SQL statements supported for data linkage.....	30
1.6 Software configuration	32
1.6.1 Software configuration for linking data from one HiRDB to another HiRDB	32
1.6.2 Software configuration for linking data from a HiRDB to a mainframe database.....	35

1.6.3	Software configuration for linking data from a mainframe database to a HiRDB	37
1.6.4	Software configuration for linking data from a mainframe database to a HiRDB using a SAM file	39
1.7	Data linkage system construction procedure	42
2.	Environment Setup	43
<hr/>		
2.1	Products associated with Datareplicator	44
2.1.1	Products required for Datareplicator	44
2.1.2	Products required for a source system	45
2.1.3	Products required for a target system	46
2.2	Installing Datareplicator (UNIX)	48
2.2.1	Preparations before installation	48
2.2.2	Server machines where Datareplicator is installed	48
2.2.3	Installing Datareplicator	49
2.2.4	Uninstalling Datareplicator	49
2.3	Directory structure (UNIX)	50
2.3.1	Directories created when Datareplicator is installed	50
2.3.2	Directory structure of a source Datareplicator	51
2.3.3	Directory structure of a target Datareplicator	54
2.4	Specifying environment variables (UNIX)	58
2.4.1	Environment variables for a source Datareplicator	58
2.4.2	Environment variables for a target Datareplicator	61
2.5	Setting up the communications environment (UNIX)	66
2.5.1	Setting up a source Datareplicator's communications environment	66
2.5.2	Setting up a target Datareplicator's communications environment	69
2.6	Installing Datareplicator (Windows)	71
2.6.1	Preparations before installation	71
2.6.2	Server machines where Datareplicator is installed	71
2.6.3	Installing Datareplicator	72
2.6.4	Information registered during installation	73
2.6.5	Post-installation procedure	74
2.6.6	Uninstalling Datareplicator	74
2.7	Directory structure (Windows)	76
2.7.1	Directories created when Datareplicator is installed	76
2.7.2	Directory structure of a source Datareplicator	77
2.7.3	Directory structure of a target Datareplicator	80
2.8	Specifying environment variables (Windows)	84
2.8.1	Environment variables for a source Datareplicator	84
2.8.2	Environment variables for a target Datareplicator	85
2.9	Setting up the communications environment (Windows)	89
2.9.1	Registering the service name	89
2.9.2	Registering the host name	90
2.9.3	Using Windows Terminal Service	90

2.10	Operating environment in Windows Vista and Windows Server 2008	92
2.10.1	Executing commands	92
2.10.2	Renamed commands	92
2.10.3	Support of JIS standard level 3 and level 4 character sets	93
2.11	Upgrading Datareplicator	94
2.11.1	Notes about upgrading.....	94
2.11.2	How to upgrade	94
2.11.3	Notes on upgrading	94
2.12	Improving the reliability of syslogfile and character encoding conversion (Linux only)	95
2.12.1	Improving the reliability of syslogfile.....	95
2.12.2	Character encoding conversion on syslogfile.....	95
2.12.3	Notes.....	96

3. Data Linkage Facilities 99

3.1	Linkage patterns	100
3.1.1	Linking data to a table with the same format	100
3.1.2	Linking data to a table with a different format.....	100
3.1.3	Linking data from one table to n tables.....	102
3.1.4	Linking data from n tables to one table.....	102
3.1.5	Linking data by selecting the rows to be sent	103
3.1.6	Linking data by using a user own coding routine	104
3.1.7	Obtaining a record of updates in chronological order.....	105
3.2	Source Datareplicator's extraction processing.....	106
3.2.1	Overview of extraction processing.....	106
3.2.2	Files and processes used during extraction processing	107
3.2.3	Units of extraction processing.....	113
3.2.4	Collecting information about the source Datareplicator	113
3.2.5	Controlling the number of transmission processes	114
3.2.6	Processing update information with a user own coding routine	116
3.2.7	Suppressing message output.....	116
3.2.8	Linking update data for concatenation operations	117
3.2.9	Data linkage for RDAREAs using the inner replica facility	118
3.3	Target Datareplicator's import processing	124
3.3.1	Overview of import processing	124
3.3.2	Files and processes used during import processing	125
3.3.3	Import methods	130
3.3.4	Import method for the multi-FES facility	133
3.3.5	Units of import processing	133
3.3.6	Data linkage for a table with a trigger set	134
3.3.7	Collecting time-ordered information.....	135
3.3.8	Collecting information about the target Datareplicator.....	136
3.3.9	Processing update information with a user own coding routine	137
3.3.10	Skipping import errors	137

3.3.11	Creating a merge table	141
3.3.12	Specifying the synchronization point processing for import processing..	143
3.3.13	Suppressing message output.....	144
3.3.14	Processing in the event of a lock error	145
3.4	Using JP1/Cm2 for operations management	146
3.4.1	Overview of using JP1/Cm2 for operations management	146
3.4.2	Status monitoring.....	148
3.4.3	Information collection	149
3.4.4	Remote control	152
3.4.5	Files and processes used for operations management	154
3.4.6	Initializing operations management.....	154
3.4.7	Operation of the supervisor machine.....	157
3.4.8	Operation of a supervised machine	162
3.4.9	MIB file	162
3.5	Datareplicator file system areas	172
3.5.1	Purpose of a Datareplicator file system area	172
3.5.2	Creating a Datareplicator file system area.....	174
3.5.3	Structure of a Datareplicator file system area.....	175
3.5.4	Notes on using Datareplicator file system areas.....	179
3.6	Delay monitoring facility.....	180
3.6.1	Overview of the delay monitoring facility	180
3.6.2	Using the delay monitoring facility	182
3.6.3	Notes.....	183
3.7	Import transaction synchronization facility	186
3.7.1	Overview of the import transaction synchronization facility	186
3.7.2	Preparations for the import transaction synchronization facility.....	186
3.7.3	Flow of processing by the import transaction synchronization facility.....	191
3.7.4	Transaction branch information.....	193
3.7.5	Synchronous import group	195
3.7.6	How to check the extraction and import status.....	199
3.7.7	Error handling.....	200
3.7.8	Notes.....	201
3.7.9	Examples	203
3.8	Event facility.....	206
3.8.1	Issuing events	207
3.8.2	Types of events	208
3.8.3	Defining event codes	208
3.8.4	Event detection timing and action	209
3.8.5	Notes.....	213
3.9	Duplexing files.....	215
3.9.1	Files that can be duplexed	216
3.9.2	Files associated with duplexing.....	217
3.9.3	Notes about duplexing	218

4.1	System design items	222
4.2	Designing a linkage pattern	224
4.2.1	Data linkage to a table with the same format	225
4.2.2	Data linkage to a table with a different format.....	229
4.2.3	Data linkage from one table to n tables.....	234
4.2.4	Data linkage from n tables to one table.....	238
4.2.5	Data linkage by selecting rows to be sent	240
4.2.6	Data linkage using a UOC routine	242
4.2.7	Acquisition of a record of update information over time.....	244
4.3	Designing the correspondence between source and target databases.....	247
4.3.1	Conditions for the source database.....	247
4.3.2	Creating a table subject to import processing	250
4.3.3	Designing the correspondence of mapping keys.....	250
4.3.4	Designing for the supported data types	253
4.3.5	Designing for character code sets.....	262
4.3.6	Designing repetition columns.....	280
4.3.7	Creating a time-ordered information table	282
4.4	Designing the correspondence between source and target systems	285
4.4.1	Designing data linkage from one HiRDB to another HiRDB	285
4.4.2	Designing data linkage from a HiRDB to a mainframe database	292
4.4.3	Designing data linkage from a mainframe database to a HiRDB	296
4.4.4	Designing data linkage from a mainframe database to a HiRDB using SAM files.....	299
4.5	Designing the data linkage system mode	302
4.5.1	Data linkage system modes	302
4.5.2	Designing a data linkage system for application to a hierarchical system..	303
4.5.3	Notes on a data linkage system that links multiple systems	305
4.6	Designing a source Datareplicator.....	313
4.6.1	Source Datareplicator's file organization	313
4.6.2	Preparation of the files used with the source Datareplicator.....	315
4.6.3	Designing the extraction procedure.....	322
4.6.4	Designing the transmission procedure	325
4.6.5	Designing the extraction processing start method.....	331
4.6.6	Designing the extraction processing stop method.....	333
4.6.7	Designing the event control table.....	335
4.6.8	Designing the source Datareplicator's resources.....	339
4.7	Designing a target Datareplicator	364
4.7.1	Target Datareplicator's file organization	364
4.7.2	Preparation of the files used with the target Datareplicator	365
4.7.3	Designing the import procedure	370
4.7.4	Designing the import processing start method.....	381
4.7.5	Designing the import processing stop method	384
4.7.6	Designing the switching of import processing methods	388

4.7.7 Designing the target Datareplicator's resources.....	389
4.8 Designing the source HiRDB	411

5. Definitions 413

5.1 Overview of Datareplicator definitions	414
5.1.1 Organization of the Datareplicator definitions	414
5.1.2 Definition rules	419
5.2 Extraction system definition	424
5.2.1 Format.....	424
5.2.2 Modifying defined information	425
5.2.3 Explanation of the operands	428
5.3 Extraction environment definition	450
5.3.1 Format.....	450
5.3.2 Modifying defined information	450
5.3.3 Explanation of the operands	452
5.4 Transmission environment definition	460
5.4.1 Format.....	460
5.4.2 Modifying defined information	461
5.4.3 Explanation of the operands	462
5.5 Extraction definition	475
5.5.1 Structure and format	475
5.5.2 Modifying defined information	476
5.5.3 Extraction definition statement.....	476
5.5.4 Transmission definition statement.....	479
5.6 Source HiRDB definition.....	485
5.6.1 System common definition.....	485
5.6.2 Unit control information definition	486
5.7 Duplexing definition (source).....	487
5.8 Import system definition	489
5.8.1 Format.....	489
5.8.2 Modifying defined information	490
5.8.3 Explanation of the operands	492
5.9 Import environment definition.....	509
5.9.1 Format.....	509
5.9.2 Modifying defined information	510
5.9.3 Explanation of the operands	513
5.10 Import definition	537
5.10.1 Definition rules	537
5.10.2 Structure and format	538
5.10.3 Modifying defined information	540
5.10.4 Update information field definition.....	540
5.10.5 Import table definition	544
5.10.6 Import group definition	551
5.11 Update information definition.....	560

5.11.1	Structure and format	560
5.11.2	Operating environment definition statement	561
5.11.3	Extraction redefinition statement	561
5.11.4	Extraction statement	563
5.12	Duplexing definition (target)	566
5.13	Examples of Datareplicator definitions	567
5.13.1	Examples of system configuration and file organization	567
5.13.2	Examples of source Datareplicator definitions	569
5.13.3	Examples of target Datareplicator definitions	573
5.13.4	Examples of source and target definitions	576
5.13.5	Examples of import group definitions when the multi-FES facility is used	583

6. Operation 589

6.1	Overview of data linkage systems	590
6.1.1	Normal data linkage system operating procedure	590
6.1.2	Data linkage from a mainframe database to a HiRDB using SAM files.....	593
6.2	Initialization procedure at the environment configuration stage	595
6.3	Startup and termination of the source Datareplicator	600
6.3.1	Starting the source Datareplicator	600
6.3.2	Terminate the source Datareplicator.....	603
6.4	Operation of the source Datareplicator.....	605
6.4.1	Handling of extraction processing.....	605
6.4.2	Handling of the files used with the source Datareplicator	610
6.4.3	Notes on handling the source Datareplicator	620
6.5	Handling of the source HiRDB	623
6.5.1	Starting, stopping, and cancelling HiRDB Datareplicator linkage	623
6.5.2	Data linkage file	624
6.5.3	Handling of the system log file	624
6.5.4	Processing at the source HiRDB and source Datareplicator depending on the specification of extsuppress in the extraction environment definition	630
6.5.5	Checking the execution status of HiRDB Datareplicator linkage.....	632
6.5.6	Source HiRDB handling procedure.....	632
6.5.7	Notes on handling when syncterm=true is specified.....	651
6.5.8	Notes on handling when sendcontrol=sendmst is specified.....	651
6.6	Startup and termination of the target Datareplicator	653
6.6.1	Starting the target Datareplicator.....	653
6.6.2	Terminate the target Datareplicator	655
6.7	Operation of the target Datareplicator	658
6.7.1	Handling of import processing.....	658
6.7.2	Handling of the files used with the target Datareplicator.....	665
6.7.3	Notes on handling the target Datareplicator.....	673
6.8	Changing the configuration of HiRDB and Datareplicator	677
6.8.1	Changing the definition of tables subject to linkage	678

6.8.2	Changing the configuration of the source system	683
6.8.3	Changing the configuration of the target system	686
6.8.4	Changing the configuration when the import transaction synchronization facility is used.....	689
6.9	Using the system switchover facility	695
6.9.1	System switchover facility modes	696
6.9.2	Preparations for using the system switchover facility (for HA monitor) ...	698
6.9.3	Preparations for using the system switchover facility (for Microsoft Cluster Server)	708
6.9.4	Handling procedure when the system switchover facility is used (for HA monitor).....	712
6.9.5	Notes on using the system switchover facility	712
6.9.6	Using the standby-less system switchover (effects distributed) facility.....	715
6.10	Using the file duplexing function	726
6.10.1	Changing the file organization from non-duplexing to duplexing	726
6.10.2	Changing the files to be duplexed	727
6.10.3	Changing the file organization from duplexing to non-duplexing	729
6.11	Handling of large files.....	731
6.11.1	Preparations for handling large files (UNIX edition only)	731
6.11.2	Estimating the command execution time when large files are used	734
6.12	Tuning	737
6.12.1	Whether tuning is needed	737
6.12.2	How to tune	738
6.13	Notes about operation	741
6.13.1	Notes about changing the OS time	741

7. Command Syntax 743

Overview of commands	745
hdechgstatus (change the status of the source Datareplicator)	752
hdeevent (issue an event at the source Datareplicator).....	753
hdefcopy (copy the current source file)	755
hdefstate (display the status of duplexed source files)	756
hdemodq (modify the organization of extraction information queue files).....	758
hdeprep (create an extraction definition preprocessing file).....	762
hdeshmclean (delete the source Datareplicator's shared resources)	764
hdestart (start the source Datareplicator)	770
hdestart_n (partially start the source Datareplicator).....	783
hdestate (collect source Datareplicator status information).....	789
hdestop (terminate the source Datareplicator)	799
hdestop_n (partially terminate the source Datareplicator).....	803
hdsagtopt (manipulate Datareplicator agent settings).....	807
hdsagtstart (start the Datareplicator agent)	809
hdsagtstatus (display the Datareplicator agent status)	810
hdsagtstop (terminate the Datareplicator agent)	812

hdscnvedt (edit a mapping table for converting character codes)	813
hdshgstatus (change the status of the target Datareplicator).....	819
hdsfcopy (copy the current target file)	820
hdsfmkfs (initialize a Datareplicator file system area).....	821
hdsfstate (display the status of duplexed target files).....	824
hdsfstafs (display the status of a Datareplicator file system area).....	826
hdspathlist (specify a directory to be monitored)	829
hdsrefinfm (check update information)	831
hdsrftcl (control import processing)	837
hdssamqin (extract update information from a SAM file)	841
hdsshmclean (delete the target Datareplicator's shared resources).....	843
hdsstart (start the target Datareplicator)	846
hdsstate (collect target Datareplicator status information).....	854
hdsstop (terminate the target Datareplicator)	865
hdstrcredit (edit an activity trace file).....	870
pdlogchg (modify the status of log-related files).....	890
pdls (display the HiRDB system status).....	891
pdrplstart (start HiRDB Datareplicator linkage)	893
pdrplstop (cancel HiRDB Datareplicator linkage)	894

8. User Own Coding Routines 895

8.1 Import information editing UOC routine	896
8.1.1 Overview of an import information editing UOC routine.....	896
8.1.2 Creating an import information editing UOC routine (UNIX)	900
8.1.3 Creating an import information editing UOC routine (Windows)	908
8.1.4 Syntax for the functions used with an import information editing UOC routine	911
8.1.5 Handling of abstract data types by an import information editing UOC routine	928
8.1.6 Notes on creating an import information editing UOC routine	937
8.1.7 Sample import information editing UOC routine	939
8.2 Column data editing UOC routine.....	944
8.2.1 Overview of a column data editing UOC routine	944
8.2.2 Column data editing UOC routine creation procedure (UNIX).....	946
8.2.3 Column data editing UOC routine creation procedure (Windows).....	949
8.2.4 Syntax for the function used with a column data editing UOC routine	952
8.2.5 Notes on creating a column data editing UOC routine	962
8.2.6 Sample column data editing UOC routine	963
8.3 Send data UOC routine.....	964
8.3.1 Overview of a send data UOC routine	964
8.3.2 Send data UOC routine creation procedure (UNIX).....	966
8.3.3 Send data UOC routine creation procedure (Windows).....	970
8.3.4 Syntax for the function used with a send data UOC routine	972
8.3.5 Notes on creating a send data UOC routine	981

8.3.6	Sample send data UOC routine	983
9.	Error Handling Procedures	985
9.1	Error handling procedures for the source Datareplicator.....	986
9.1.1	Error handling procedures	986
9.1.2	Error handling methods	988
9.1.3	Actions after correcting an error.....	992
9.1.4	Procedures for handling errors at the target system.....	993
9.1.5	User own coding routine error handling procedure.....	995
9.1.6	Handling of file errors during file-duplexed operation	995
9.2	Error handling procedures for the target Datareplicator	996
9.2.1	Error handling procedures	996
9.2.2	Error handling methods	998
9.2.3	Actions after correcting an error.....	1001
9.2.4	Procedures for handling errors at the source system	1002
9.2.5	User own coding routine error handling procedure.....	1003
9.3	Error recovery method selection criteria	1004
9.4	Initialization procedure during error recovery	1008
9.4.1	Errors that require initialization of Datareplicators	1008
9.4.2	Datareplicator initialization procedure during error recovery	1009
9.5	Data linkage recovery via the system log file.....	1012
9.5.1	Overview of data linkage recovery via the system log file	1012
9.5.2	Prerequisites for data linkage recovery via the system log file	1014
9.5.3	Overview of the recovery procedure using the system log file	1018
9.6	Data linkage recovery via unload log files	1023
9.6.1	Overview of data linkage recovery via unload log files.....	1023
9.6.2	Prerequisites for data linkage recovery via unload log files.....	1024
9.6.3	Preparing for data linkage recovery via unload log files.....	1029
9.6.4	Overview of the recovery procedure using unload log files.....	1030
9.6.5	Details of data linkage recovery using unload log files.....	1036
9.6.6	Commands provided by the data linkage recovery facility	1048
9.6.7	Procedure after execution of data linkage recovery facility	1053
9.6.8	How to suppress import when the table-based import method is used for import processing at the target system.....	1054
9.7	Facility for recovering the extraction information queue.....	1058
9.7.1	Overview of the facility for recovering the extraction information queue file.....	1058
9.7.2	Prerequisites for the facility for recovering the extraction information queue file.....	1059
9.7.3	Recovery procedure using the facility for recovering the extraction information queue file	1061
9.8	Acquisition of untransmitted information due to import errors (update-SQL output facility).....	1066

10. Messages	1077
10.1 Overview of messages	1078
10.1.1 Message output destination	1078
10.1.2 Message output format	1078
10.1.3 Message descriptive format.....	1080
10.2 Details about messages	1085
10.3 List of system call errors	1337
10.4 List of cause codes.....	1340
Appendixes	1347
A. Detailed Information About HiRDB-Related Datareplicator Support	1348
A.1 OSs supported for HiRDB and the versions of Datareplicator that can be used on those OSs.....	1348
A.2 HiRDB facilities and their support by Datareplicator	1350
B. Datareplicator Reserved Words	1362
C. Functional Differences Between the UNIX and Windows Editions of Datareplicator	1366
D. Downgrading Datareplicator	1368
D.1 Differences in downgrading procedures between product models.....	1368
D.2 Downgrading procedure	1368
D.3 Notes after downgrading	1368
E. Glossary	1370
Index	1379

Chapter

1. Overview

This chapter describes the features of Datareplicator, data linkage, the software configuration, and the procedure for constructing a data linkage system.

- 1.1 Features
- 1.2 Data linkage
- 1.3 Tables supported for data linkage
- 1.4 Data types for tables supported for data linkage
- 1.5 SQL statements supported for data linkage
- 1.6 Software configuration
- 1.7 Data linkage system construction procedure

1.1 Features

HiRDB Datareplicator (referred to hereafter simply as *Datareplicator*) automatically imports update information for another database into a database in a HiRDB system. It also imports update information from a HiRDB system database into another system's database.

1.1.1 Purpose of Datareplicator

This section discusses the purpose of Datareplicator.

(1) *Problems of distributed databases*

Large volumes of data created by corporate activities have been accumulating in databases on mainframe systems. Recently, many companies have been distributing data from their mainframe systems to workstations (WSs) and personal computers (PCs) by employing a database management system (DBMS), such as HiRDB, that runs on workstations or personal computers. As this type of system operation becomes increasingly popular, questions of how to integrate and utilize distributed databases become important issues.

(2) *Overview of replication facilities*

The facilities for importing data from a distributed database into another database are called *replication facilities*. Replication facilities enable you to import data between databases in different systems, thereby providing support for data management in a distributed system environment.

There are two types of replication facilities:

Data linkage facility

You use the data linkage facility to extract data from a database that has been updated and to automatically import the database update information into another system's database. This process of automatically importing data into another system's database is called *data linkage*. A database system equipped with the data linkage facility is called a *data linkage system*.

The data linkage facility is useful when you use the most recent data of one database in another system's database, and when you automatically back up databases.

Database extraction/import service facility

You use the database extraction/import service facility to extract data from a database in the batch mode and to import data into a database in another system. The database extraction/import service facility is useful when you extract and store a large amount of data in batch mode, and when you create and re-create

databases in other systems.

The following table lists the products required for using the replication facilities with HiRDB and mainframe databases.

Table 1-1: Products required for using the replication facilities

Replication facility	HiRDB data linkage product	Data linkage product for mainframe database
Data linkage facility	HiRDB Datareplicator	XDM/DS ^{#2}
Database extraction/import service facility	HiRDB Dataextractor ^{#1}	XDM/XT ^{#3}

#1

For details about HiRDB Dataextractor, see the *HiRDB Dataextractor Version 8 Description, User's Guide and Operator's Guide*.

#2

For details about XDM/DS, see the *VOS3 XDM/DS* manual.

#3

For details about XDM/XT, see the *VOS3 XDM/XT* manual.

(3) Advantages of a Datareplicator data linkage system

One way of linking data between two databases is to update one database whenever the other database is updated. The drawback to this method is that it requires a very heavy system workload for synchronization, error recovery, and so on. In contrast, data linkage using Datareplicator reduces the system workload because it extracts update information from one database, and then uses that data for asynchronous updating of a different database in another system.

Data linkage using Datareplicator is effective when performance and reduction of overall system workload are more important than precise synchronization of the distributed databases.

Datareplicator's data linkage facility also provides the following benefits:

- There is no need to modify applications, because Datareplicator extracts update information automatically.
- The data linkage facility transfers only the updated data, thereby minimizing the amount of transfer data and the key system's workload.
- You can create customized tables tailored to the needs of the various data-warehouse users, because the data linkage facility allows you to, among other things, select the tables and columns that will be subject to update

processing and accumulate update information in chronological order.

(4) Terminology for Datareplicator's data linkage systems

The following table defines the terminology for data linkage systems used in this manual.

Table 1-2: Terminology for data linkage systems

Terminology for data linkage system	Definition
Source system	The system from which the updated data to be imported is extracted (linkage source)
Source Datareplicator	Datareplicator running in the source system
Source database #	Database in the source system
Target system	The system into which the extracted update data is imported (linkage target)
Target	Datareplicator running in the target system
Target database #	Database in the target source system

#

To identify a source database or target database by the system in which it is running, a HiRDB database is called a *source HiRDB database* or a *target HiRDB database*, as appropriate, while a database at a mainframe system is called a *mainframe database*.

1.1.2 Data linkage system

(1) Systems whose data can be linked

HiRDB enables the following combinations of data linkage:

- Data linkage from one HiRDB database to another HiRDB database
- Data linkage from a HiRDB database to a mainframe database
- Data linkage from a mainframe database to a HiRDB database
- Data linkage from a mainframe database to a HiRDB database by using SAM files

For details about the supported data linkage combinations, see *1.2.1 Combinations for a data linkage system*.

(2) Application systems to which data linkage is applicable

You can apply data linkage to the following types of applications:

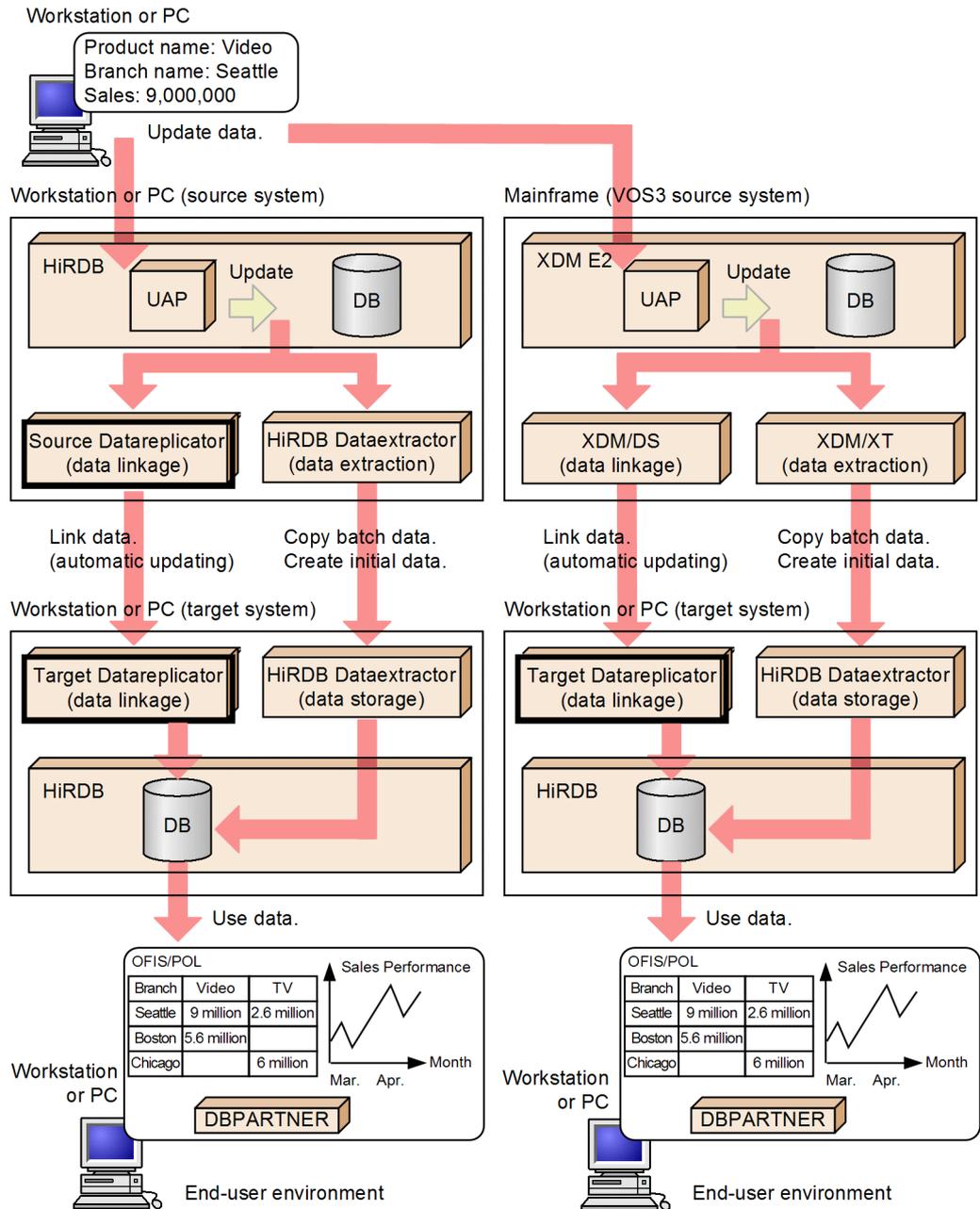
- Data exchange between key applications that are run in different departments

(constructed as a mainframe or a server)

- Data addition to a system that stores a history of key applications
- Data extraction from key applications in order to create departmental systems
- Data exchange between departmental systems
- Extraction of a departmental system's backup data

The following figure shows an example of an application system to which data linkage is applied.

Figure 1-1: Example of application system to which data linkage is applied

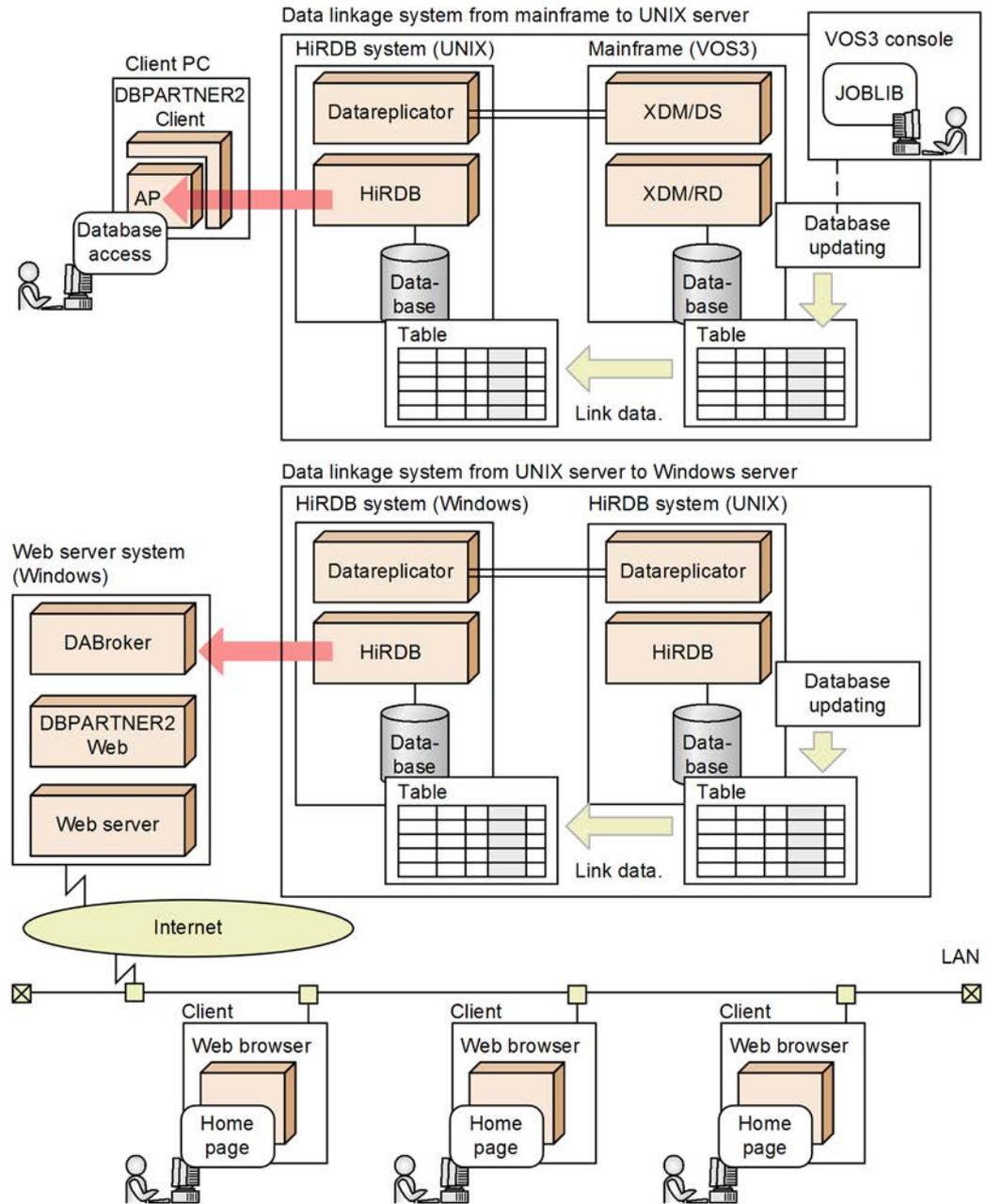


(3) Application to data warehouses

HiRDB's replication facilities enable you to not only extract/import data between HiRDBs but also to utilize mainframe databases (legacy data with strong asset elements) with HiRDB. Taking advantage of these replication facilities, you can construct an integrated database system linking multiple databases (*data warehouse*). With a data warehouse, you can use online analytical processing (OLAP) tools to analyze a database system and enable end users to access from PCs data that is linked to a HiRDB.

The figure below provides an example of applying HiRDB to a data warehouse using the replication facilities. This example uses Datareplicator to link key applications' databases and to reference their data from personal computers. It omits data warehouse elements, such as database system analysis and decision-making systems.

Figure 1-2: Example of applying HiRDB to a data warehouse using the replication facilities



1.2 Data linkage

A data linkage system consists of the source Datareplicator and the target Datareplicator or these products plus XDM/DS.

1.2.1 Combinations for a data linkage system

Table 1-3 shows the combinations of source and target databases that can achieve data linkage using Datareplicator.

Table 1-3: Combinations of source and target databases that can achieve data linkage using Datareplicator

Source database (applicable OSs#)	Target database (applicable OSs#)
HiRDB (UNIX, Windows)	HiRDB (UNIX, Windows) XDM/RD E2 (VOS3)
XDM/SD E2 (VOS3)	HiRDB (UNIX, Windows)
XDM/RD E2 (VOS3)	
ADM (VOS3)	
PDM2 E2 (VOS3, VOS1)	
TMS-4V/SP (VOS3)	
RDB1 E2 (VOS1)	

Note:

For details about the combinations of mainframe systems for implementing data linkage, see the *VOS3 XDM/DS* manual or the applicable database manual.

#

Each database product is applicable under the indicated operating systems only.

1.2.2 Mechanism of data linkage

This section explains the mechanism of data linkage for each combination of data linkage systems.

(1) Data linkage from one HiRDB to another HiRDB

This section explains the mechanism of data linkage from one HiRDB to another HiRDB.

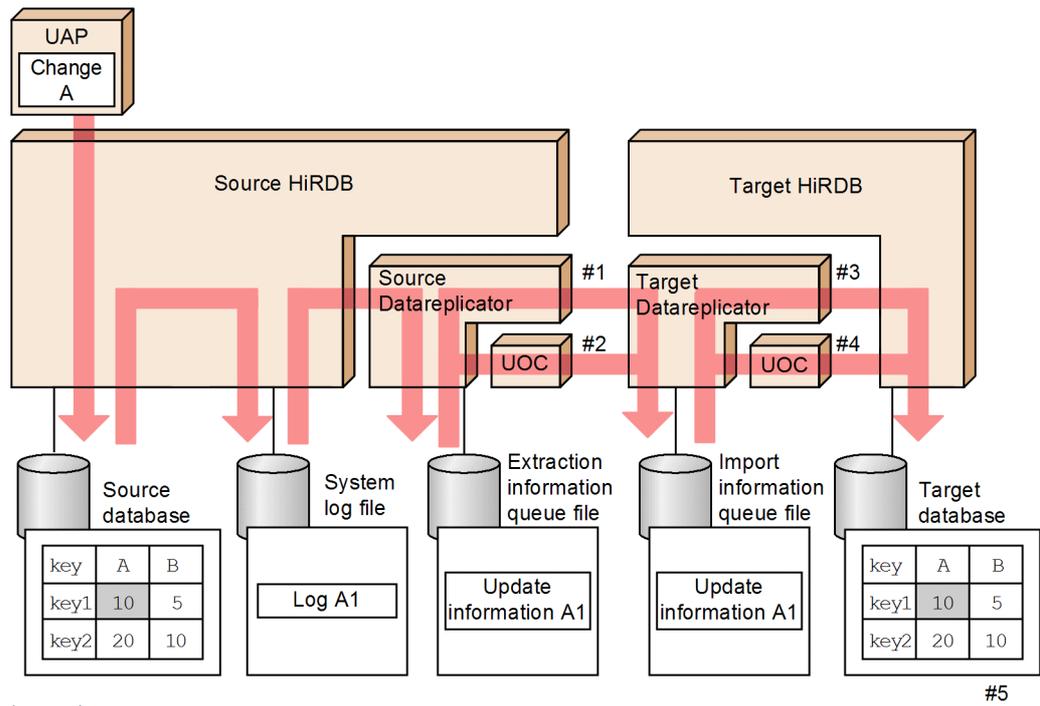
When the source database is updated, the source Datareplicator extracts the update information (information used by the UAP to update the database) from the HiRDB's

system log file and stores it in an extraction information queue file. This is called *extraction processing*. You can use a user own coding (UOC) routine to check the update information in order to determine whether to send it to the target Datareplicator.

The update information is sent to the target Datareplicator at the interval defined in the source Datareplicator and is stored in an import information queue file, from where it is imported into the target database in the transaction units defined in the target Datareplicator. This is called *import processing*. Note that this update processing at the target database is not synchronized with the update processing in the source database.

Update information can be imported into the target database by the target Datareplicator either automatically or at a time of your choosing by using a UOC routine. Figure 1-3 shows the mechanism of data linkage from one HiRDB into another HiRDB.

Figure 1-3: Mechanism of data linkage from one HiRDB to another HiRDB



#1

The source Datareplicator sends the data automatically.

#2

This UOC routine checks the update information, and then sends it.

#3

The target Datareplicator imports the data automatically.

#4

This UOC routine edits the update information, and then imports it.

#5

The target Datareplicator imports the data automatically. A UOC routine can be used to process the update information before importing it into the target database.

(2) Data linkage from a HiRDB to a mainframe database

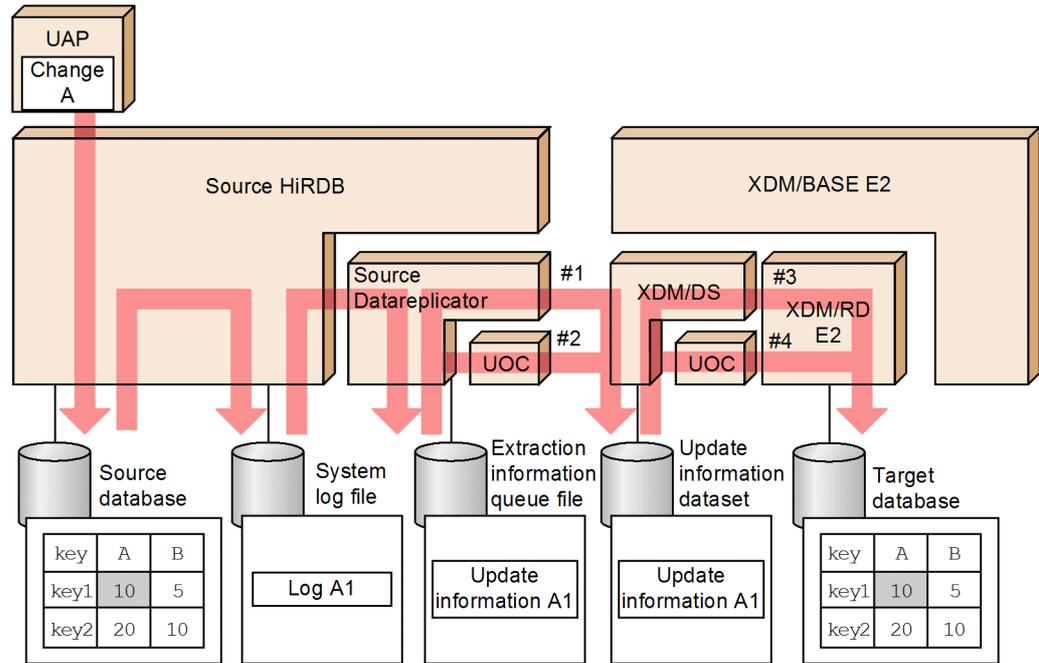
This section explains the mechanism of data linkage from a HiRDB to a mainframe database (XDM/RD E2).

When the source database is updated, the source Datareplicator extracts the update information from the HiRDB's system log file and stores it in an extraction information queue file. A UOC routine can be used to check the update information and determine whether to send it to the target Datareplicator.

The update information is sent to the XDM/DS data linkage product at the mainframe system at the interval defined in the source Datareplicator and is stored in the update information dataset. The stored update information is imported into the target database in the transaction units of the source database defined in XDM/DS. Note that this update processing at the target database is not synchronized with the update processing in the source database.

Update information can be imported into the target database by XDM/DS either automatically or at a time of your choosing by using a UOC routine. The following figure shows the mechanism of data linkage from a HiRDB to a mainframe database.

Figure 1-4: Mechanism of data linkage from a HiRDB to a mainframe database



Legend:

■ : Subject to extraction or import

#1

The source Datareplicator sends the data automatically.

#2

This UOC routine checks the update information, and then sends it.

#3

The target XDM/DS imports the data automatically.

#4

This UOC routine edits the update information, and then imports it.

(3) Data linkage from a mainframe database to a HiRDB

This section explains the mechanism of data linkage from a mainframe database (XDM/SD E2, XDM/RD E2, ADM, PDMII E2, or TMS-4V/SP) to a HiRDB.

When the source database is updated, the XDM/DS data linkage product on the mainframe system extracts the update information and stores it in an update

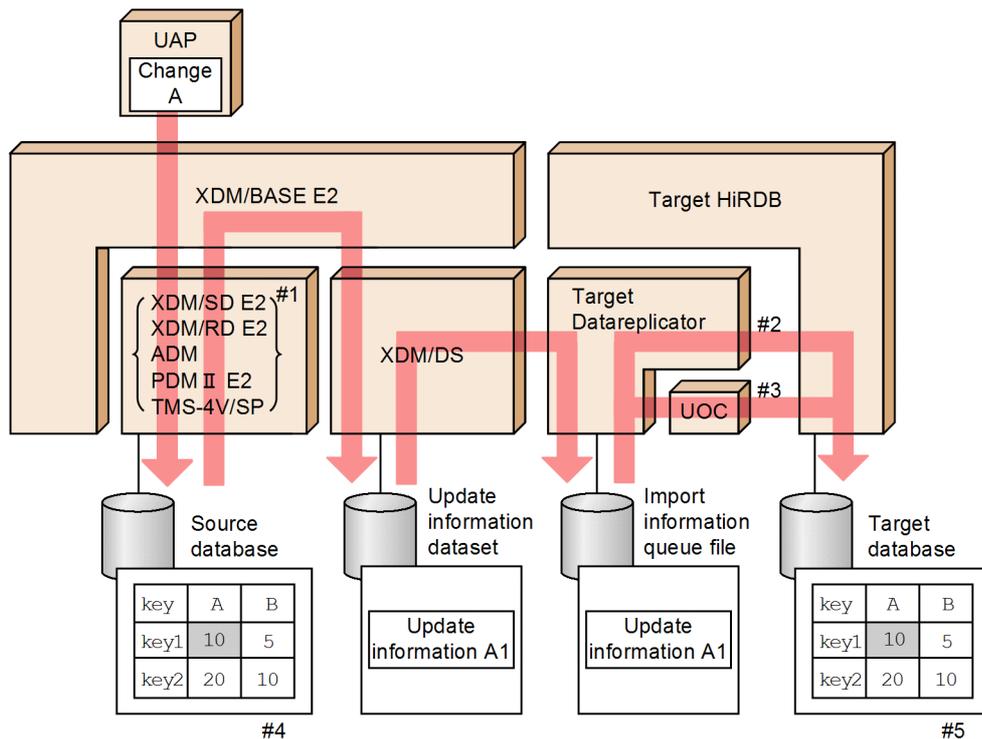
information dataset.

The update information is sent to the target Datareplicator at the interval specified by XDM/DS and stored in an import information queue file. The update information is then imported into the target database in the source database transaction units defined in the target Datareplicator. Note that update processing for the target database is not synchronized with update processing for the source database.

Update information can be imported into the target database by Datareplicator either automatically or at a time of your choosing by using a UOC routine.

The following figure shows the mechanism of data linkage from a mainframe database to a HiRDB.

Figure 1-5: Mechanism of data linkage from a mainframe database to a HiRDB



Legend:

■ : Subject to extraction or import

#1

One of XDM/SD E2, XDM/RD E2, ADM, VOS3 PDMII E2, or TMS-4V/SP is required. For TMS-4V/SP, the TMS-4V/SP/Data Linkage Support is required.

#2

The target Datareplicator imports the data automatically.

#3

This UOC routine edits the update interface, and then imports it.

#4

The source database is XDM/RD E2.

#5

The target Datareplicator imports the data automatically. A UOC routine can be used to process the update information before importing it into the target database.

(4) Data linkage from a mainframe database to a HiRDB using a SAM file

A SAM file stores data sequentially from the beginning of the file. Datareplicator uses SAM files to link data in VOS1 PDMII E2 or VOS1 RDB1 E2. This subsection explains the mechanism of data linkage from a mainframe database (PDMII E2 or RDB1 E2) to a HiRDB using a SAM file.

When the source database is updated, the source Datareplicator extracts the update information on the basis of definitions in the mainframe system, and then stores it in a SAM file via the mainframe's update information dataset. This SAM file is then transferred in binary format to the target Datareplicator by one of the following file transfer programs:

VOS3 PDM2 E2

File transfer using VOS3 XFIT, XNF/TCP, or IFIT-TSS E2

VOS1 PDM2 E2 or VOS1 RDB1 E2

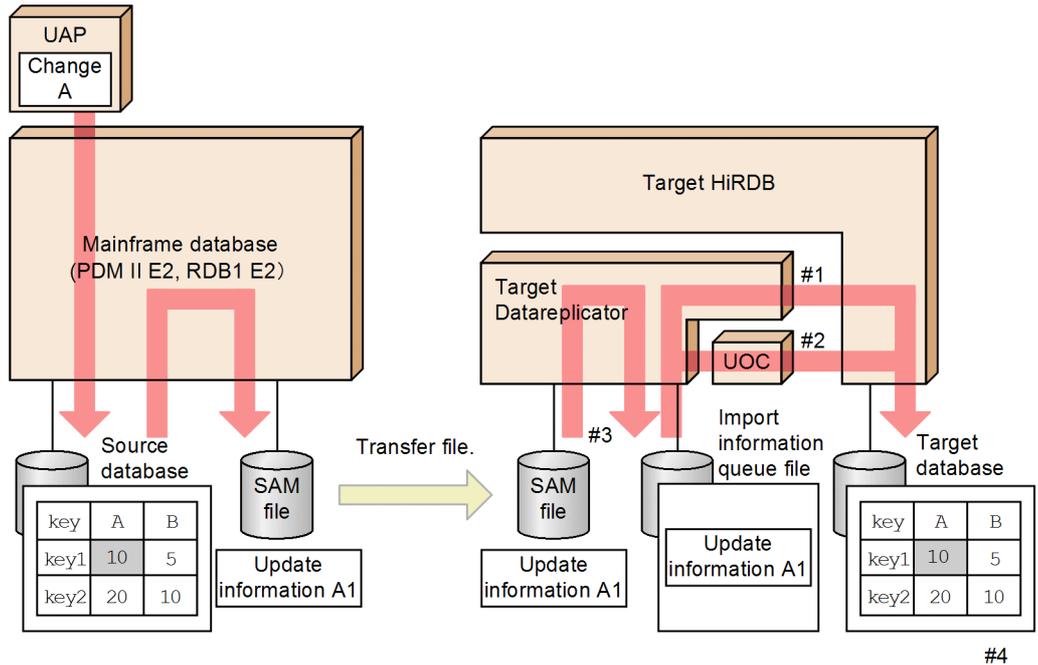
File transfer using VOS1 XFIT, HCAM/TCP, or IFIT/IEX

The directory used to store the SAM file in the target system depends on a specification for the file transfer program.

You must execute manually the target Datareplicator's update information input command (`hdssamqin` command) in order to store the update information from the SAM file into an import information queue file. The update information stored in the import information queue file is imported into the target database in the source database transaction units defined in the target Datareplicator. Note that this update processing at the target database is not synchronized with the update processing in the source database.

Update information can be imported into the target database by Datareplicator either automatically or at a time of your choosing by using a UOC routine. The following figure shows the mechanism of data linkage from a mainframe database to a HiRDB using a SAM file.

Figure 1-6: Mechanism of data linkage from a mainframe database to a HiRDB using a SAM file



Legend:

■ : Subject to extraction or import

#1

The target Datareplicator imports the data automatically.

#2

This UOC routine edits the update interface, and then imports it.

#3

To store update information from a SAM file into an import information queue file, use the Datareplicator's `hdssamqin` command (execute this command manually).

#4

The target Datareplicator imports the data automatically. A UOC routine can be used to process the update information before importing it into the target database.

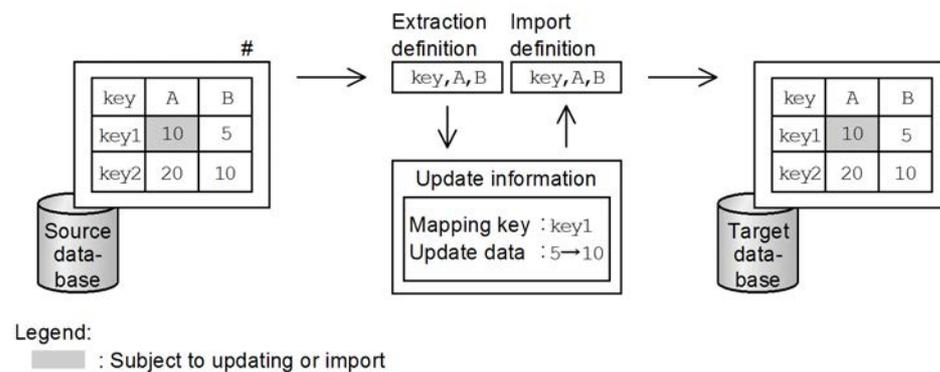
1.2.3 Correspondence between source and target databases

To link data between source and target databases, you must establish data correspondence between the source and target databases that use the update information. Information about the source database is called the *extraction definition*, while information about the target database is called the *import definition*.

You must specify these definitions for the data linkage product (Datareplicator or XDM/DS). When you use a SAM file in linking data, you must specify the mainframe database definitions before you create the dataset to be transferred to the file transfer program.

You use the extraction definition to establish the correspondence between the source database and the update information, and you use the import definition to establish the correspondence between the update information and the target database. The update information consists of mapping keys that identify specific rows with specific update data. The following figure shows the correspondence between the source and target databases.

Figure 1-7: Correspondence between the source and target databases



#

This example assumes that a HiRDB is the source database.

1.2.4 Databases supported for data linkage

This section describes the databases that can be linked using Datareplicator.

(1) Types of databases

The types of databases supported for data linkage by Datareplicator include structured-type databases (structured-type databases, hierarchical-type databases, and network-type databases) and relational-type databases. In this manual, structured-type databases are referred to as *structured databases* and relational-type databases are referred to as *relational databases*.

The following table shows the types of databases and the relationship between the source and target databases.

Table 1-4: Types of databases supported for data linkage and the relationship between the source and target databases

Source system		Target system	
Type	Source database	Type	Target database
Relational	HiRDB XDM/RD E2 TMS-4V/SP# RDB1 E2	Relational	HiRDB
Structured	XDM/SD E2		XDM/RD E2
Hierarchical	ADM		
Network	PDM2 E2		

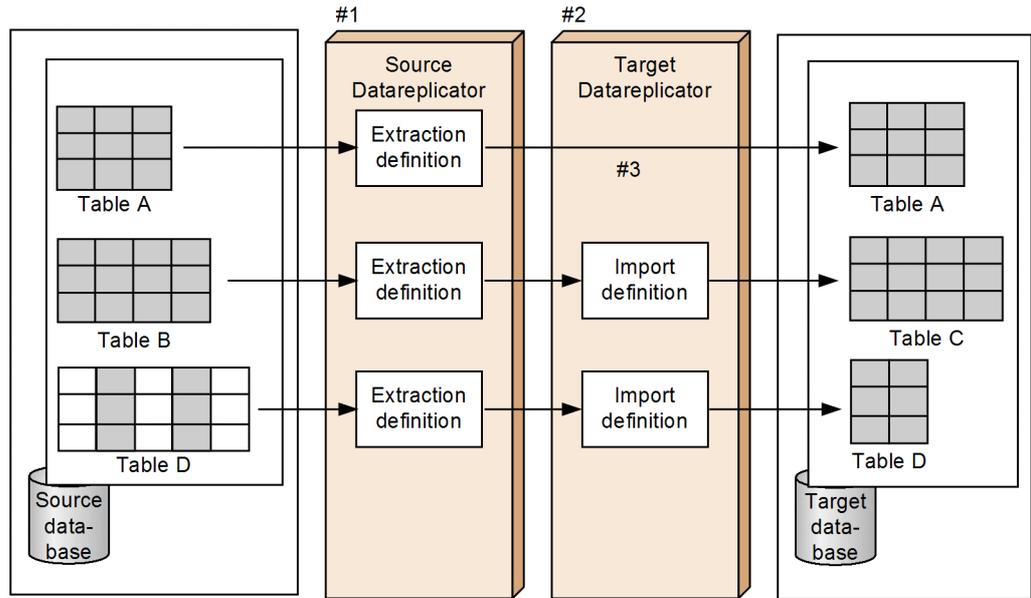
#

Although TMS-4V/SP uses hierarchical-type data, it is classified as a relational database because TMS-4V/SP/Data Linkage Support outputs data with a relational structure.

(2) Relationship between database type and data linkage

Figure 1-8 shows the concept of data linkage between relational databases, and Figure 1-9 shows the concept of data linkage from a structured database to a relational database.

Figure 1-8: Concept of data linkage between relational databases



Legend:

[shaded box] : Subject to extraction or import

#1

The source database is a HiRDB. If the source database is XDM/RD E2, XDM/DS is required instead of a source Datareplicator.

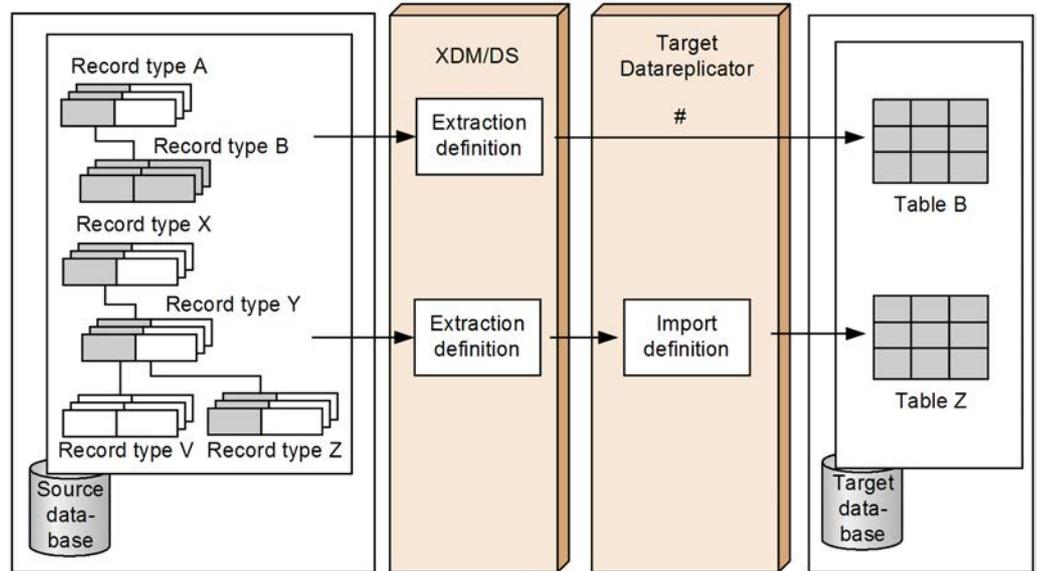
#2

The target database is a HiRDB. If the target database is XDM/RD E2, XDM/DS is required instead of a target Datareplicator.

#3

The import definition can be omitted when a table in the source database is imported into an identical table in the target database (one that has the same table format, table name, column names, and so on).

Figure 1-9: Concept of data linkage from a structured database to a relational database



Legend:

■ : Subject to extraction or import

#

If the import definition is omitted, the target database's *authorization-identifier.table-identifier* corresponds to the source database's *schema.record-type*.

1.2.5 Correspondence of database terminology

If the source and target databases are of different types, they use different database terminology. Table 1-5 shows the correspondence of database terms. This manual uses the database terminology for relational databases.

Table 1-5: Correspondence of database terms

Structured, hierarchical, and network databases			Relational database
XDM/SD E2	ADM	PDM2 E2	HiRDB, XDM/RD E2, TMS-4V/SP#, or RDB1 E2
Schema	Database	DBM	database
record type	Segment	dataset	table

1. Overview

Structured, hierarchical, and network databases			Relational database
XDM/SD E2	ADM	PDM2 E2	HiRDB, XDM/RD E2, TMS-4V/SP#, or RDB1 E2
record occurrence	segment occurrence	record	row
Component	Field	field	column

#

Although TMS-4V/SP uses hierarchical-type data, it is classified as a relational database because TMS-4V/SP/Data Linkage Support outputs data with a relational structure.

1.3 Tables supported for data linkage

The table below lists the tables supported for data linkage by Datareplicator. The information in this table applies to both the source and the target Datareplicator.

Classification		Data linkage support
User table	No option specified.	Y
	SHARE option specified.	R ^{#1}
	WITHOUT ROLLBACK option specified.	R ^{#2}
	INSERT ONLY option specified.	R ^{#3}
	INNER CONSTRUCTOR option specified.	R ^{#4}
	CHARACTER SET option specified.	R ^{#5}
	Triggers are defined.	R ^{#6}
	COMPRESSED option specified.	R ^{#7}
	Other options are specified.	Y
View table		N
Data dictionary table		N
Temporary table		N

Legend:

Y: Supported.

R: Supported with restrictions.

N: Not supported.

#1

Data linkage is supported only for the back-end server that was specified in the server name operand in the `create rdarea` statement when a shared RDAREA for storing shared tables was created. (Shared tables are tables that have the `SHARE` option specified.) To apply data linkage to shared tables, you must specify the `with lock` clause in the `load` statement in the target Datareplicator's import definition. There is no need to specify this definition in the source Datareplicator. For details about the import definitions, see *5.10.5 Import table definition*.

#2

There are prerequisites and limitations when data linkage is applied to tables for which the `WITHOUT ROLLBACK` option specified. For details, see *6.7.3(4) Notes about tables for which the `WITHOUT ROLLBACK` option is specified.*

#3

Use the same structure for both source and target tables.

#4

Data linkage is supported if a column with this option specified is not extracted.

#5

When a column with the `CHARACTER SET` option specified is to be extracted, both the conditions described below must be satisfied. If either of these conditions is not satisfied, inconsistencies might occur between the source and target databases.

- The same `CHARACTER SET` option is specified for the source and the target columns.
- If the source database is `XDM/DS` and `EBCDIK` is specified in the `CHARACTER SET` option for a target column, `nocodecny` is specified for the field corresponding to that column by using the `format` statement in the import definition.

#6

There are prerequisites and limitations when data linkage is applied to tables for which triggers are defined. For details, see *3.3.6 Data linkage for a table with a trigger set.*

#7

There are prerequisites and limitations when data linkage is applied to tables for which the `COMPRESSED` option is specified. For details, see *6.7.3(5) Notes about tables for which the `COMPRESSED` option is specified.*

1.4 Data types for tables supported for data linkage

The data types supported by Datareplicator are the same as those supported by HiRDB. However, some data types are converted to data types that can be identified by Datareplicator. You can also apply data linkage to HiRDB's special data types (such as abstract data type columns and repetition columns).

1.4.1 Data types supported by Datareplicator

The following table lists the data types supported by Datareplicator.

Table 1-6: Data types supported by Datareplicator

Classification	Data type of column subject to data linkage
Numeric type	INTEGER, SMALLINT, DECIMAL, LARGEDECIMAL, FLOAT, SMALLFLT
Character type	CHAR ^{#3} , VARCHAR ^{#3} , NCHAR, NVARCHAR, MCHAR, MVARCHAR
Date type	DATE, TIME, TIMESTAMP, INTERVAL YEAR TO DAY, INTERVAL HOUR TO SECOND
Abstract data type ^{#1}	SGMLTEXT, FREEWORD, XML
Special type ^{#2}	BLOB, BINARY

Note 1:

Datareplicator performs data linkage even on columns that have not been updated as if those columns have been updated with the same values. However, data linkage is not performed on columns of the following data types if they have not been updated:

- VARCHAR type (256 bytes or greater)
- NVARCHAR type (128 characters or greater)
- MVARCHAR type (256 bytes or greater)
- BLOB type
- BINARY type (256 bytes or greater)
- Abstract data type (SGMLTEXT, FREEWORD, XML)

Note 2:

- Datareplicator can extract data, even when a repetition column is created for a data type. For details about data linkage for a table containing repetition columns, see *1.4.3 Data linkage for a table containing repetition columns*.

- LONG VARCHAR, LONG NVARCHAR, and LONG MVARCHAR are recognized as VARCHAR, NVARCHAR, and MVARCHAR, respectively.

Note 3:

To perform data linkage on the characters of JIS standard levels 3 and 4, some conditions must be satisfied. For details, see *2.10.3 Support of JIS standard level 3 and level 4 character sets*.

#1

Some rules apply to data linkage on abstract data types. For details, see *1.4.2 Data linkage for a table using an abstract data type*.

#2

Some rules apply to data linkage on BLOB and BINARY types. For details, see *1.4.4 Data linkage for tables using BLOB and BINARY type columns*.

#3

Data linkage can be performed even when a character set is specified, provided that the same character set is specified for the source and the target columns. For details about character sets, see the description of EBCDIC in *4.3.5 Designing for character code sets*.

When the target system is a HiRDB, the data type of a column extracted by Datareplicator is imported as the same data type. For details about the data types when data is imported into XDM/RD, see the *VOS3 XDM/DS* manual and the applicable database manual. For details about the data type relationships between the source/target systems and Datareplicator, see *4.3.4 Designing for the supported data types*.

1.4.2 Data linkage for a table using an abstract data type

Datareplicator can link data in columns of a HiRDB abstract data type (ADT). An *abstract data type* refers to a type of large data with a complex structure, such as multimedia data. HiRDB enables multimedia data to be stored in a table as an abstract data type and accessed with SQL statements, in the same manner as with conventional database tables.

The following shows an example of a table containing the SGMLTEXT type.

- SQL statement for creating a table containing the SGMLTEXT type (CREATE TABLE)

```
CREATE TABLE REPORT (TITLE CHAR(32),DATE DATE,AUTHOR CHAR(32),
                     DOCUMENT SGMLTEXT RECOVERY ALL ALLOCATE (sgmltext IN(PDUSER02))
                     PLUGIN '<TEXTTYPE>SGML</TEXTTYPE><DTD>MAN.dtd</DTD>
                     <NORparam>MANnorm.prm</NORparam>'
                     ) IN (PDUSER01);
```

- Report table created by CREATE TABLE (SGMLTEXT-type column: DOCUMENT)

Column data type	CHAR(32)	DATE	CHAR(32)	SGMLTEXT
Column name	TITLE	DATE	AUTHOR	DOCUMENT
	:	:	:	:

(1) Abstract data types supported for data linkage by Datareplicator

The abstract data types supported for data linkage by Datareplicator include SGMLTEXT, FREEWORD, and XML. Data linkage is supported for abstract data types only between HiRDB data linkage systems.

(2) Correspondence of data types between source and target systems

To link data with an abstract data type, the source and target systems must satisfy the following conditions:

- The source and target databases are both HiRDB.
- The columns at the source and the target must both be of the abstract data type.
- The definition of the abstract data type (such as type name, number of attributes, and attribute data type) must be identical at the source and target.
- The abstract data type's owner (authorization identifier) must be identical at the source and target.
- The definition of the constructor function for the abstract data type must be identical at the source and target.
- Only one argument is passed to a constructor function.
- The definitions specific to the HiRDB plug-in's abstract data type must be identical at the source and target.
- The document type definitions match between the source and the target (applies only to the SGMLTEXT type).

- The character encoding matches between the source and the target (applies only to the XML type).

Note:

Datareplicator does not check that these conditions are satisfied. The target HiRDB's data compatibility checking can determine whether valid data has been imported. If data import into the target HiRDB results in an error, Datareplicator outputs an error message and terminates import processing. The error handling procedure is the same as when compatibility checking on existing data types results in an error.

1.4.3 Data linkage for a table containing repetition columns

Datareplicator allows you to link data in HiRDB repetition columns. A *repetition column* is a column that contains multiple elements per cell. With HiRDB, you can store a table column consisting of multiple elements per cell as a repetition column and access it by means of SQL statements, in the same manner as with conventional database tables.

- SQL statement for creating a table that contains repetition columns (CREATE TABLE)

```
CREATE TABLE STAFF_TABLE (NAME NVARCHAR(10),
                           SEX NCHAR(1),
                           FAMILY NVARCHAR(5) ARRAY[10],
                           RELATIONSHIP NVARCHAR(5) ARRAY[10]);
```

- STAFF_TABLE created by CREATE TABLE (repetition columns: FAMILY and RELATIONSHIP)

Column data types	NVARCHAR(10)	NCHAR(1)	NVARCHAR(5)	NVARCHAR(5)
Column names	NAME	SEX	FAMILY	RELATIONSHIP
	○×△◇	MALE	○○○○○ ××××× : △△△△△ ◇◇◇◇◇	FATHER MOTHER : BROTHER SISTER
	:	:	:	:
	:	:		FATHER MOTHER : BROTHER SISTER

Note that Datareplicator can link data in units of a repetition column's cells, but not in units of its elements.

(1) Correspondence of data types between source and target systems

To link data in a repetition column, the source and target systems must satisfy the following conditions:

- The repetition column must be defined at the source and target.
- The definitions of the repetition column (type name, number of attributes, attribute data type, number of element data items, and sequence of element data items) must be identical at the source and target.

Even when all these conditions are not satisfied, data linkage might be possible if an import information editing UOC routine is used, because it can edit the data.

Note:

Datereplicator does not check that these conditions are satisfied. The target HiRDB's data compatibility checking can determine whether valid data has been imported. If data import into the target HiRDB results in an error, Datereplicator outputs an error message and terminates import processing. The error handling procedure is the same as when compatibility checking on existing data types results in an error.

1.4.4 Data linkage for tables using BLOB and BINARY type columns

Datereplicator can perform data linkage on BLOB and BINARY type data, subject to certain rules. This subsection explains the rules for BLOB and BINARY type data.

(1) Rules for BLOB type columns

The rules for performing data linkage on BLOB-type columns are as follows:

- To perform data linkage on BLOB-type columns, specify ALL in the RECOVERY operand in HiRDB's table definition. For details about HiRDB table definition, see the manual *HiRDB Version 9 SQL Reference*.
- Datereplicator uses the BLOB column definition lengths, not the actual lengths of the BLOB data, to perform processing (including buffer allocation). For this reason, use for the definition lengths of BLOB columns values that are as close as possible to the actual data lengths.
- A definition error will occur if an attempt is made to extract a BLOB-type column whose definition length is 2 GB or greater or whose definition length was omitted during table creation (in which case HiRDB assumes 2 GB as the definition length). We recommend that you use for data linkage BLOB-type columns whose definition length is a maximum of several megabytes.

If a BLOB-type column has a definition length of 2 GB or greater but its actual data is small, you can perform data linkage on such a column by specifying the HDE_BIN_COL_MAXLEN environment variable in the source Datereplicator without having to re-define the table. For details about the

HDE_BIN_COL_MAXLEN environment variable, see the following subsections:

- 2.4.1 *Environment variables for a source Datareplicator (UNIX)*
- 2.8.1 *Environment variables for a source Datareplicator (Windows)*
- 6.4.3(4) *Notes about the HDE_BIN_COL_MAXLEN environment variable*
- BLOB-type source columns cannot be used as mapping keys.
- Data linkage is not performed on a BLOB-type column that is subject to processing by an HiRDB utility even if the utility's processing outputs update logs. This is because Datareplicator does not recognize data in BLOB-type columns updated by utilities. Note that the results of utility execution on BLOB-type columns are not imported into the target database.
- For unimported BLOB-type data in the target Datareplicator, update data is output in the following format:

*BLOB(*data-length*)*

(2) Rules for BINARY type columns

The rules for performing data linkage on BINARY-type columns are as follows:

- Taking into account performance and memory requirements, we recommend that you perform data linkage on BINARY-type columns whose definition length is no more than 10 MB.
- BINARY-type source columns cannot be used as mapping keys.
- For unimported BINARY-type data in the target Datareplicator, update data is output in the following format:

*BINARY(*data-length*)*

(3) Backward deletion updating for the BLOB and BINARY types

You can perform backward deletion updating on BLOB and BINARY types if you specify BACKWARD_CUTOFF_UPDATE in the pd_rpl_func_control operand in HiRDB's system common definition. For details about backward deletion updating on BLOB and BINARY types, see the *HiRDB Version 9 UAP Development Guide*.

The following limitations apply to backward deletion updating for BLOB and BINARY types:

- Data cannot be imported to merge tables.
- Data cannot be imported to chronologically-ordered information tables.
- Column data editing UOC routines cannot be used.
- Send data UOC routines cannot be used.

Notes about specifying the pd_rpl_func_control operand:

If you will not be performing backward deletion updating on BLOB or BINARY types, do not specify `BACKWARD_CUTOFF_UPDATE` in the `pd_rpl_func_control` operand in HiRDB's system common definition.

Before you change the `BACKWARD_CUTOFF_UPDA` specification in the `pd_rpl_func_control` operand, make sure that all update logs have been imported. If the `pd_rpl_func_control` operand is changed while there is an unimported update log, problems such as unmatched SQL statement execution counts between the source and target databases occur.

1.5 SQL statements supported for data linkage

The following table lists and describes the SQL statements that are supported for data linkage by Datareplicator.

Table 1-7: SQL statements supported for data linkage by Datareplicator

SQL type	Supported for data linkage	Limitations
INSERT	Y	Data linkage is not supported for a table for which the <code>WITHOUT ROLLBACK</code> option is specified.
UPDATE	Y	Data linkage is also performed for any column that is not specified in the <code>SET</code> clause of the <code>UPDATE</code> statement (a column that is not updated) as if it has been updated by the same value. However, data linkage is not performed for an unupdated column if it has any of the following data types: <ul style="list-style-type: none"> • <code>VARCHAR</code> type (256 bytes or greater) • <code>NVARCHAR</code> type (128 characters or greater) • <code>MVARCHAR</code> type (256 bytes or greater) • <code>BLOB</code> type • <code>BINARY</code> type • Abstract data type (<code>SGMLTEXT</code>, <code>FREWORD</code>, <code>XML</code>)
DELETE	Y	Data linkage cannot be performed for a table for which the <code>WITHOUT ROLLBACK</code> option is specified.
PURGE	Y	<ul style="list-style-type: none"> • Data linkage cannot be performed for a table for which the <code>WITHOUT ROLLBACK</code> option is specified. • Data linkage cannot be performed when a table subject to extraction is divided and placed in multiple BESs. For details about the processing when the <code>PURGE</code> statement is executed on a table that has been divided and placed in multiple BESs, see the <code>prg_eventno</code> operand in <i>5.4 Transmission environment definition</i>.
Other SQL statement	N	--

Legend:

Y: Data linkage is supported with some limitations.

N: Data linkage is not supported.

Note:

If multiple rows in the source database are updated by a single SQL statement, the

target Datareplicator issues as many SQL statements as there are rows updated in the source database (data linkage is performed in units of updated rows, not in units of SQL statements).

1.6 Software configuration

There are four types of system configurations for data linkage systems. This section explains the software configuration for each type.

- System configuration for linking data from one HiRDB to another HiRDB
- System configuration for linking data from a HiRDB to a mainframe database
- System configuration for linking data from a mainframe database to a HiRDB
- System configuration for linking data from a mainframe database to a HiRDB using a SAM file

When the source database is a HiRDB:

Install the source Datareplicator on the server machine that contains the source HiRDB's system manager and on the server machine that contains the HiRDB back-end server that contains the table to be extracted.

When the target database is a HiRDB:

Install the target Datareplicator on the server machine that contains the target HiRDB. If a HiRDB client is also installed on the same server machine, you can install the target Datareplicator on a different server machine from the target HiRDB.

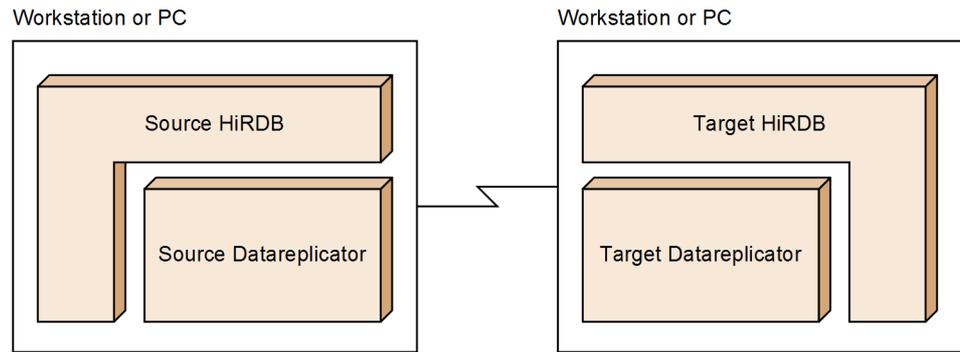
The required software programs depend on the system configuration. For details about the system configurations and required software configurations, see Section 2.1 *Products associated with Datareplicator*.

The system configurations presented here are common to both UNIX and Windows.

1.6.1 Software configuration for linking data from one HiRDB to another HiRDB

The following figure shows the software configuration for linking data from one HiRDB to another HiRDB.

Figure 1-10: Software configuration for linking data from one HiRDB to another HiRDB



(1) Correspondences between the source and target systems

The following correspondences between the source and target systems apply to linking data from one HiRDB to another HiRDB:

- Correspondence between the source HiRDB and the source Datareplicator
 Source HiRDB (single server) to source Datareplicator = 1 to 1
 Source HiRDB (parallel server) to source Datareplicator = 1 to n
- Correspondence between the source and target Datareplicators
 Source Datareplicator to target Datareplicator = 1 to m
 Source Datareplicator to target Datareplicator = n to 1
- Correspondence between the target Datareplicator and the target HiRDB
 Target Datareplicator to target HiRDB = n to 1

(2) Configurations of the source and target systems

In the case of linking data from one HiRDB to another HiRDB, the software configuration depends on whether the source HiRDB is a single server or a parallel server. Figure 1-11 shows the configurations of the systems when the source HiRDB is a single server, and Figure 1-12 shows the configurations of the systems when the source HiRDB is a parallel server.

Figure 1-11: Configurations of the source and target systems for linking data from one HiRDB to another HiRDB: Source HiRDB is a single server

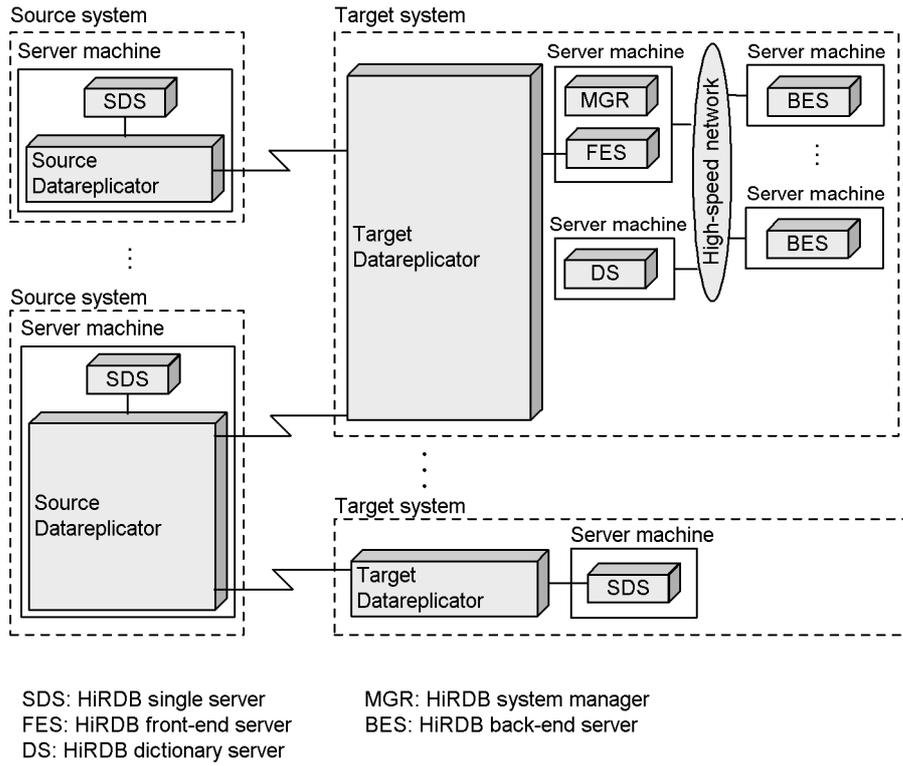
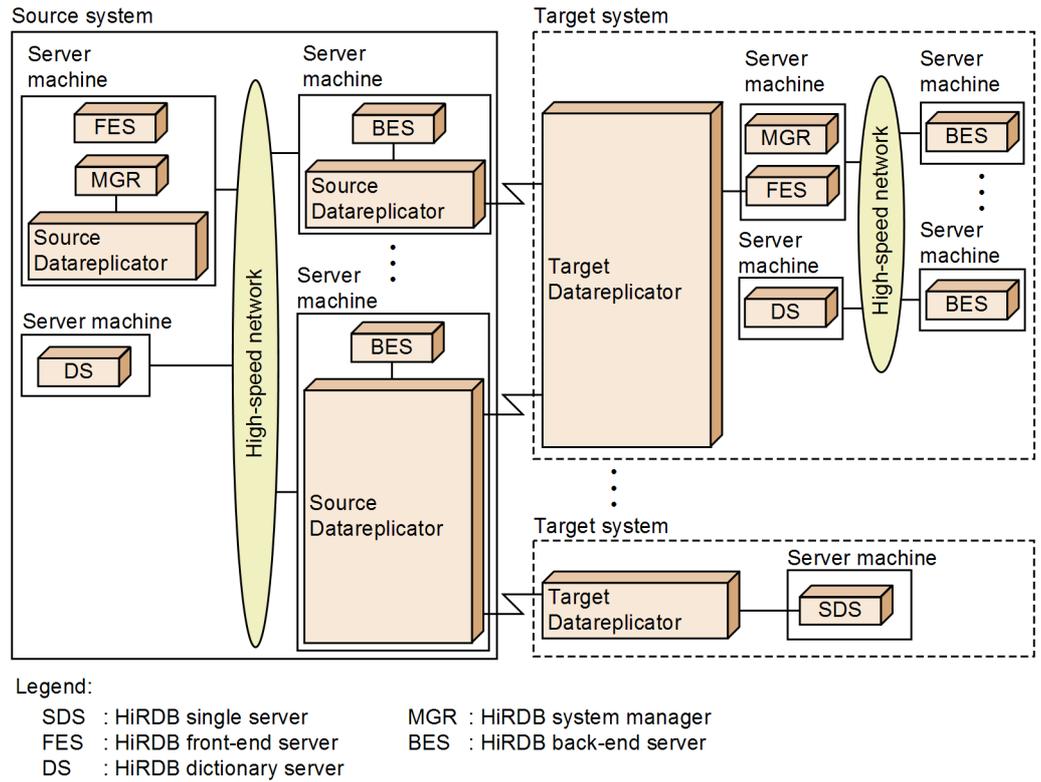


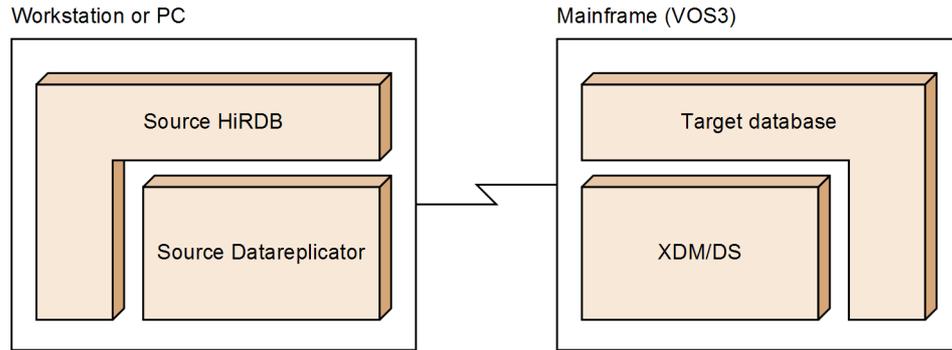
Figure 1-12: Configurations of the source and target systems for linking data from one HiRDB to another HiRDB: Source HiRDB is a parallel server



1.6.2 Software configuration for linking data from a HiRDB to a mainframe database

The following figure shows the software configuration for linking data from a HiRDB to a mainframe database (XDM/RD E2).

Figure 1-13: Software configuration for linking data from a HiRDB to a mainframe database



(1) Correspondences between the source and target systems

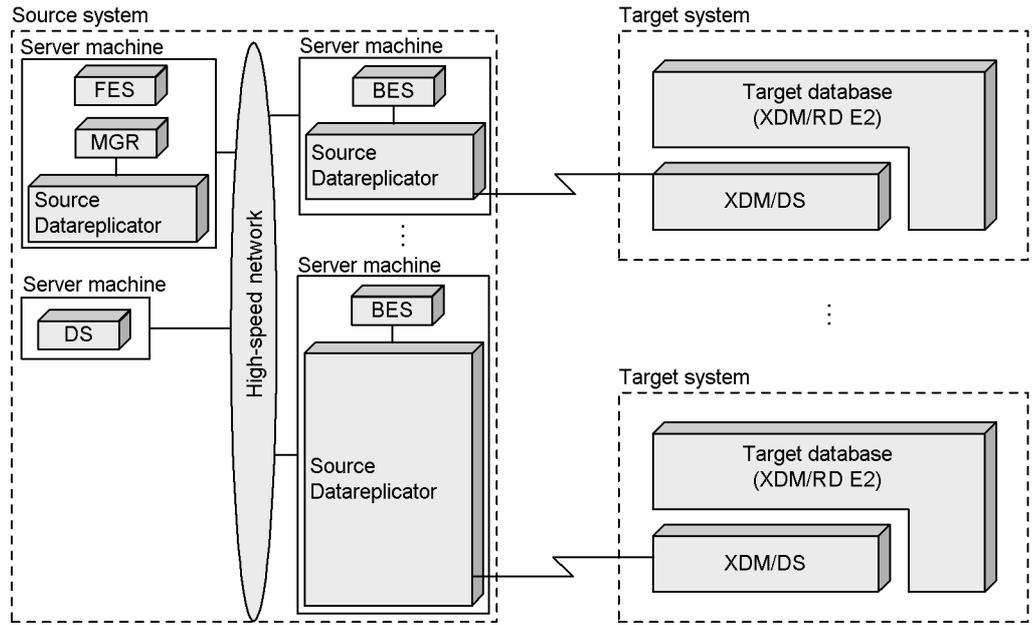
The following correspondences between the source and target systems apply to linking data from a HiRDB to a mainframe database (XDM/RD E2):

- Correspondence between the source Datareplicator and the source HiRDB
 Source HiRDB (single server) to source Datareplicator = 1 to 1
 Source HiRDB (parallel server) to source Datareplicator = 1 to n
- Correspondence between Datareplicator and XDM/DS
 Source Datareplicator to XDM/DS = 1 to m
 Source Datareplicator to XDM/DS = 1 to 1
- Correspondence between the source database and XDM/DS
 See the *VOS3 XDM/DS* manual.

(2) Configurations of the source and target systems

The following figure shows the configurations of the source and target systems for linking data from a HiRDB to a mainframe database (XDM/RD E2).

Figure 1-14: Configurations of the source and target systems for linking data from a HiRDB to a mainframe database

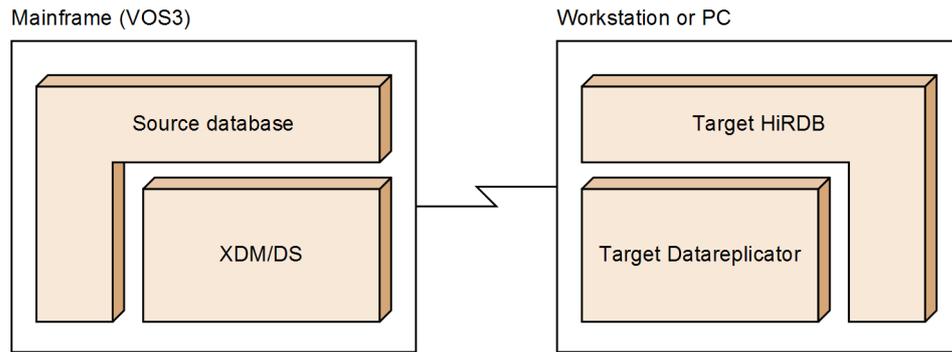


FES: HiRDB front-end server MGR: HiRDB system manager
 DS: HiRDB dictionary server BES: HiRDB back-end server

1.6.3 Software configuration for linking data from a mainframe database to a HiRDB

The following figure shows the software configuration for linking data from a mainframe database (XDM/SD E2, XDM/RD E2, ADM, PDMII E2, or TMS-4V/SP) to a HiRDB.

Figure 1-15: Software configuration for linking data from a mainframe database to a HiRDB



(1) Correspondences between the source and target systems

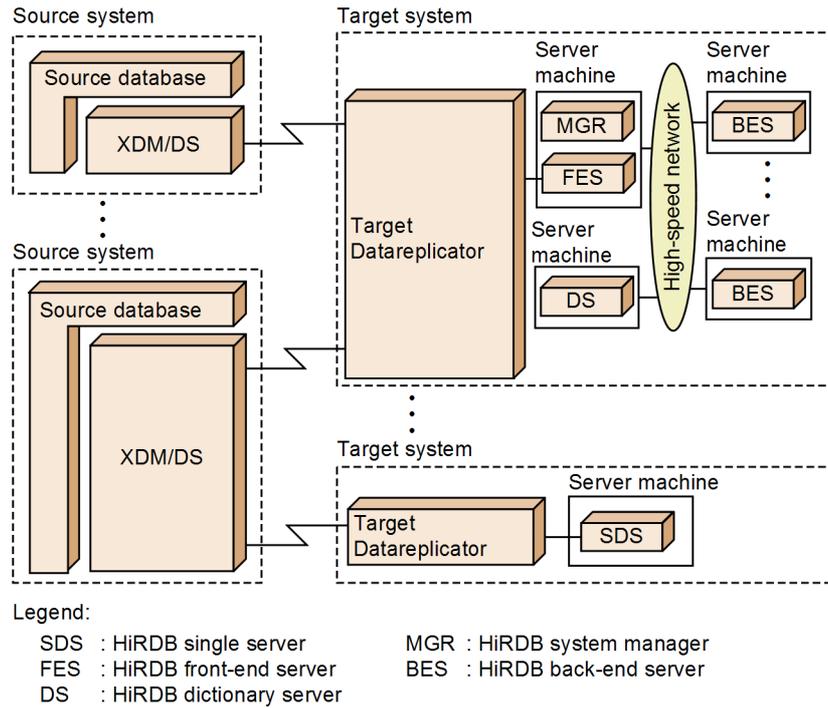
The following correspondences between the source and target systems apply to linking data from a mainframe database (XDM/SD E2, XDM/RD E2, ADM, PDMII E2, or TMS-4V/SP) to a HiRDB.

- Correspondence between the source database and XDM/DS
See the manual *VOS3 XDM Data Linkage Facility XDM/DS Description and Definition*.
- Correspondence between XDM/DS and Datareplicator
XDM/DS to target Datareplicator = 1 to m
XDM/DS to target Datareplicator = n to 1
- Correspondence between the target Datareplicator and target HiRDB
Target Datareplicator to target HiRDB = n to 1

(2) Configurations of the source and target systems

The following figure shows the configurations of the source and target systems for linking data from a mainframe database to a HiRDB.

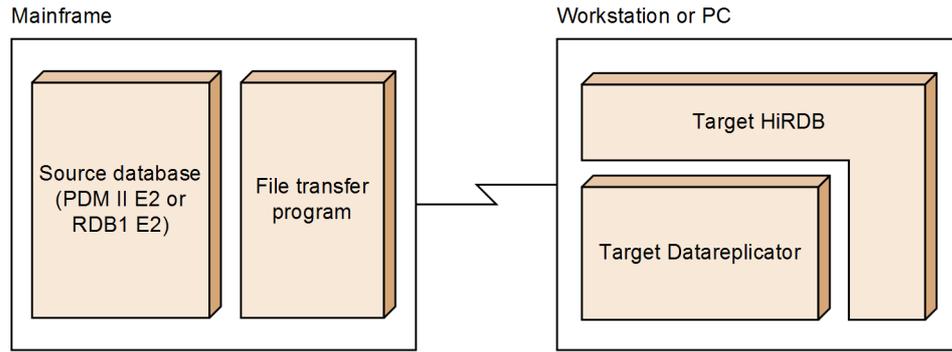
Figure 1-16: Configurations of the source and target systems for linking data from a mainframe database to a HiRDB



1.6.4 Software configuration for linking data from a mainframe database to a HiRDB using a SAM file

The following figure shows the software configuration for linking data from a mainframe database (PDMII E2 or RDB1 E2) to a HiRDB using a SAM file.

Figure 1-17: Software configuration for linking data from a mainframe database to a HiRDB using a SAM file



(1) Correspondences between the source and target systems

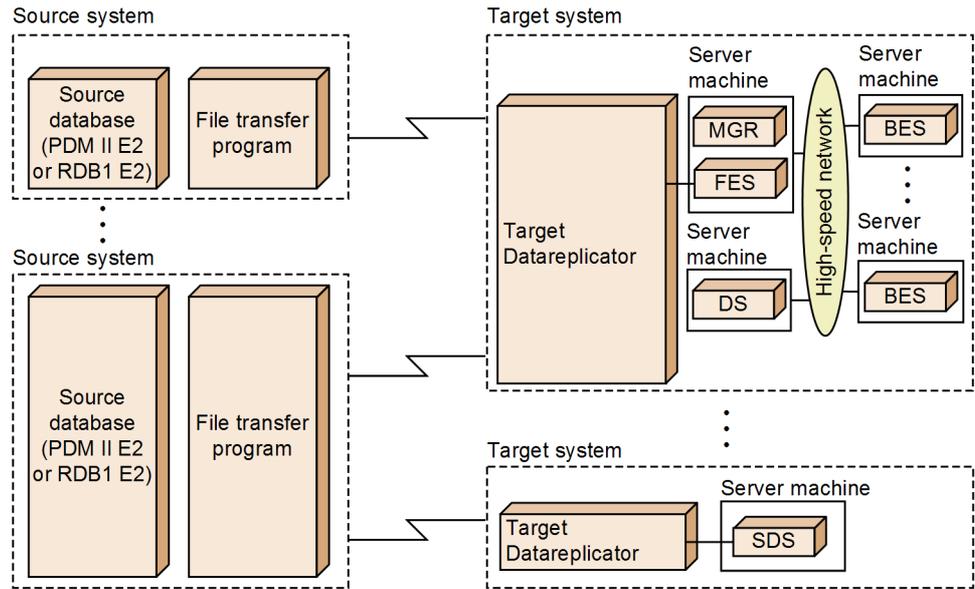
The following correspondences between the source and target systems apply to linking data from a mainframe database to a HiRDB using a SAM file:

- Correspondence between the source database and Datareplicator
There is no limit to the number of source databases or target Datareplicators.
- Correspondence between the target Datareplicator and the target HiRDB
Target Datareplicator to target HiRDB = n to 1

(2) Configurations of the source and target systems

The following figure shows the configurations of the source and target systems for linking data from a mainframe database (PDMII E2 or RDB1 E2) to a HiRDB using a SAM file.

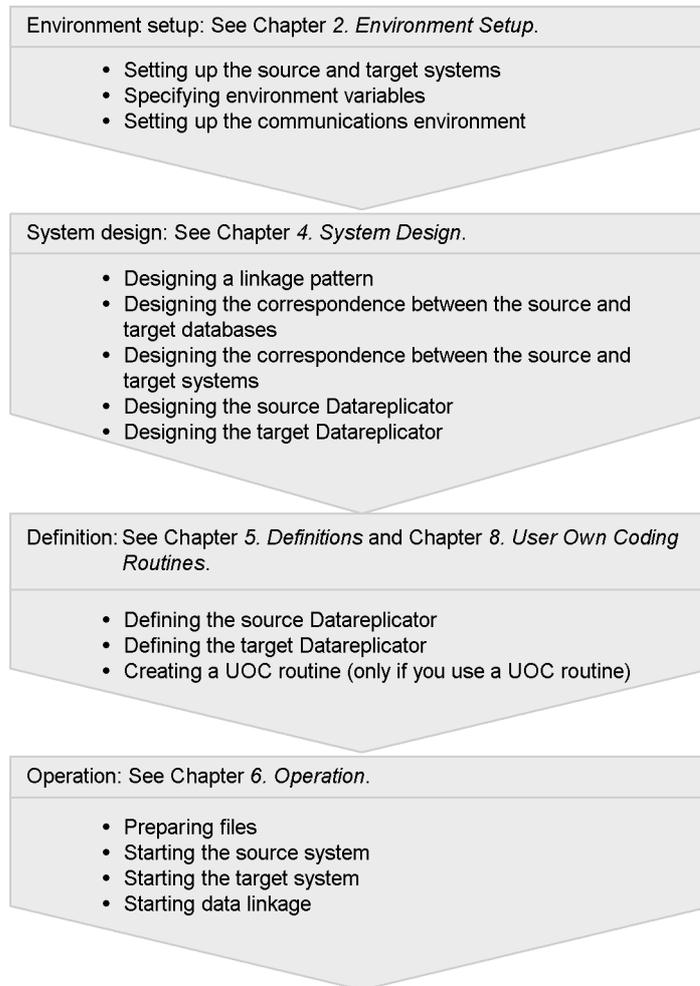
Figure 1-18: Configurations of the source and target systems for linking data from a mainframe database to a HiRDB using a SAM file



1.7 Data linkage system construction procedure

The following figure outlines the procedure for constructing a system for data linkage once you have installed Datareplicator. For details about how to construct and operate a HiRDB system, see the *HiRDB Version 9 Installation and Design Guide* or *HiRDB Version 9 System Operation Guide*.

Figure 1-19: Procedure for constructing a system for data linkage



Chapter

2. Environment Setup

This chapter describes the products associated with Datareplicator, the Datareplicator installation procedure and the directories that must be created, the specification of environment variables, and the communications environment. The environment setup procedure for UNIX Datareplicator (Sections 2.2 through 2.5) is explained separately from the procedures for Windows Datareplicator (Sections 2.6 through 2.9).

- 2.1 Products associated with Datareplicator
- 2.2 Installing Datareplicator (UNIX)
- 2.3 Directory structure (UNIX)
- 2.4 Specifying environment variables (UNIX)
- 2.5 Setting up the communications environment (UNIX)
- 2.6 Installing Datareplicator (Windows)
- 2.7 Directory structure (Windows)
- 2.8 Specifying environment variables (Windows)
- 2.9 Setting up the communications environment (Windows)
- 2.10 Operating environment in Windows Vista and Windows Server 2008
- 2.11 Upgrading Datareplicator
- 2.12 Improving the reliability of syslogfile and character encoding conversion (Linux only)

2.1 Products associated with Datareplicator

This section discusses the software products listed as follows; these products are required in order to use Datareplicator:

- Products required for Datareplicator
- Products required for a source system
- Products required for a target system

2.1.1 Products required for Datareplicator

The following table lists the products required for Datareplicator. These products are required for both the source and target Datareplicators.

Table 2-1: Products required for Datareplicator

Product name		Requirement	Remarks
OS (UNIX)	HP-UX	Y	One of these operating systems is required.
	Solaris		
	AIX 5L		
	Linux		
OS (Windows)	Windows 2000		
	Windows Server 2003		
	Windows XP		
	Windows Vista		
	Windows Server 2008		
DBMS (HiRDB)	HiRDB/Single Server	Y	HiRDB is required. The product to be installed depends on whether a HiRDB/Single Server or a HiRDB/Parallel Server is being used.
	HiRDB/Parallel Server		
	HiRDB Text Search Plug-in	N	This is a HiRDB plug-in product that is required to use columns of the abstract data type (SGMLTEXT or FREEWORD type).
	HiRDB XML Extension	N	This is a HiRDB plug-in product that is required to use columns of the abstract data type (SGMLTEXT or FREEWORD type).

Product name		Requirement	Remarks
JP1/Cm2	Cm2/Extensible SNMP Agent or JP1/Cm2/Extensible SNMP Agent	N	Required to use JP1/Cm2 for operations management.
Peripheral program products	XNF/H/BASE	N	Required to use an OSI protocol (channel connection) for communication in the HP-UX edition of Datareplicator.

Y: One is required.

N: Might be required, depending on conditions.

#

For details about the JP1/Cm2 products required for a supervisor machine (manager), see 3.4 *Using JP1/Cm2 for operations management*.

2.1.2 Products required for a source system

The following table lists the products that are required for a source system. In the table, OS and peripheral program products are omitted.

Table 2-2: List of products required for a source system

Source database	Product required for data linkage	Remarks
HiRDB	HiRDB Datareplicator	--
XDM/SD E2	XDM/DS	--
XDM/RD E2	XDM/DS	--
ADM	ADM/EB	Required in order to run ADM in an online environment.
	XDM/DS or VOS3 Database Datareplicator ^{#1}	--
PDMII E2 (VOS3)	PDMII/EF	Required in order to perform data linkage using XDM/DS or Database Datareplicator.
	XDM/DS or VOS3 Database Datareplicator ^{#1}	Data can also be linked by using a created SAM file instead of using XDM/DS or VOS3 Database Datareplicator.

2. Environment Setup

Source database	Product required for data linkage	Remarks
PDMII E2 (VOS1) ^{#2}	--	Create a SAM file in the source database system, and then use one of the following file transfer programs to transfer the SAM file to the target system: VOS1 XFIT/FTP, VOS1 HCAM/TCP, or VOS1 IFIT/IEX.
TMS-4V/SP ^{#3}	TMS-4V/SP (VOS3)	Required to link data by using TMS-4V/SP.
	XDM/DS or Database Datareplicator ^{#1}	--
RDB1 E2 ^{#2}	--	Create a SAM file in the source database system, and then use one of the following file transfer programs to transfer the SAM file to the target system: VOS1 XFIT/FTP, VOS1 HCAM/TCP, or VOS1 IFIT/IEX.

Legend:

--: No product is required for data linkage.

#1

In addition to XDM/DS, VOS3 Database Datareplicator is also a data linkage product for mainframe databases. The information to be defined in Datareplicator is the same whether the data linkage product used in VOS3 is XDM/DS or VOS3 Database Datareplicator.

#2

Because data cannot be linked automatically, the extracted data must be imported by the target Datareplicator.

#3

When TMS-4V/SP is used as the source database, the data format differs from when any other source database is used. For details about extracting data from TMS-4V/SP, see the *VOS3 XDM/DS* manual or the *VOS3 TMS-4V/SP* manual.

2.1.3 Products required for a target system

The following table lists the products that are required for a target system. In the table, OS and peripheral program products are omitted.

Table 2-3: List of products required for a target system

Target database	Product required for data linkage
HiRDB	HiRDB Datareplicator
XDM/RD E2	XDM/DS

2.2 Installing Datareplicator (UNIX)

This section explains the procedure for installing UNIX Datareplicator.

2.2.1 Preparations before installation

Before installing Datareplicator, check the following:

- Operating environment
- User privileges

(1) *Checking the operating environment*

Checking the operating environment involves the following:

- Check that the machine on which Datareplicator is to be installed has sufficient free space.

For details about the disk space required for Datareplicator, see *4.6.8 Designing the source Datareplicator's resources* or *4.7.7 Designing the target Datareplicator's resources*.

- Check that all products required for Datareplicator operation have been installed.

For details about the products required for Datareplicator operation, see *2.1 Products associated with Datareplicator*.

(2) *Checking user privileges*

Check that the person installing Datareplicator has the *superuser* user privilege.

2.2.2 Server machines where Datareplicator is installed

The server machines where Datareplicator is installed depend on whether the HiRDB system being used is a HiRDB/Single Server or a HiRDB/Parallel Server. When a HiRDB/Parallel Server is being used, the server machines depend on whether it is the source or target system.

The following table shows the server machines where Datareplicator can be installed.

Table 2-4: Server machines where Datareplicator is installed

Type of HiRDB system	Source system	Target system
HiRDB/Single Server	Install on the server machine where the single server is located.	Install on the server machine where the single server is located.
HiRDB/Parallel Server	Install on all server machines that contain the following servers: <ul style="list-style-type: none"> • System manager (MGR) • Back-end servers (BES) 	Install one Datareplicator on each server machine where the HiRDB client facility is used.

2.2.3 Installing Datareplicator

See *2.1 Products associated with Datareplicator* and install all products that are required in order to use UNIX Datareplicator. You use Hitachi Program Product Installer to install UNIX Datareplicator.

You can install HiRDB before Datareplicator, or vice versa; however, you must install HiRDB on a server machine before you can specify the Datareplicator's environment variables on that server machine.

2.2.4 Uninstalling Datareplicator

You use Hitachi Program Product Installer to uninstall UNIX Datareplicator.

2.3 Directory structure (UNIX)

When you install a UNIX edition of Datareplicator, the directory `hirdbds/` is created under the `/opt` directory, and then Datareplicator directories and files are stored at that location. To set up the Datareplicator's operating environment, you must create various definition files.

This section describes the UNIX Datareplicator's directory structure.

2.3.1 Directories created when Datareplicator is installed

Figure 2-1 shows the directories that are created when UNIX Datareplicator is installed, and Table 2-5 shows the file contents. These directories are the same for both the source and target Datareplicators.

Figure 2-1: Directories and files associated with Datareplicator

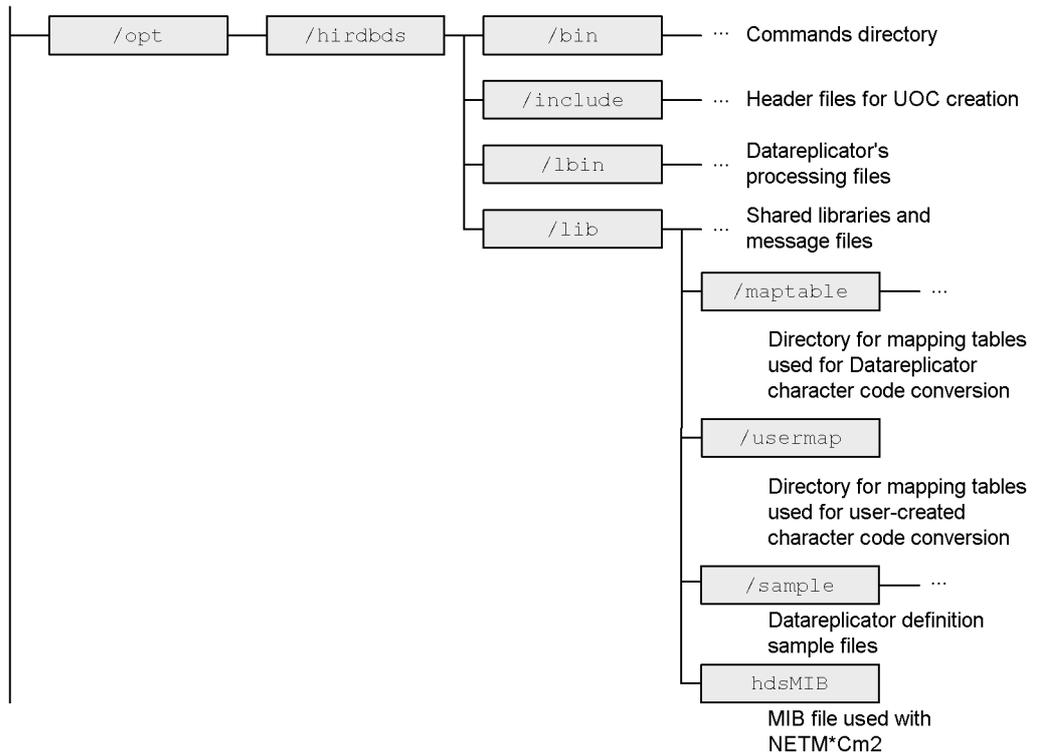


Table 2-5: Contents of directories and files created when Datareplicator is installed

Creation time	Directory and filename	Description
Installation	/opt/hirdbds/bin/	Directory containing Datareplicator commands
	/opt/hirdbds/include/	Header files for UOC creation
	/opt/hirdbds/sbin/	Datareplicator's processing files
	/opt/hirdbds/lib/	Datareplicator's shared libraries and message files
	/opt/hirdbds/lib/maptable/	Directory for mapping tables used for Datareplicator character code conversion
	/opt/hirdbds/lib/usermap/	Directory for mapping tables used for user-created character code conversion
	/opt/hirdbds/lib/sample/	Sample Datareplicator definition and UOC files ^{#1}
	/opt/hirdbds/lib/hdsMIB	MIB file used with JP1/Cm2 ^{#2}

#1: For details about the filenames for the Datareplicator definition template, see *5.13.2 Examples of source Datareplicator definitions* or *5.13.3 Examples of target Datareplicator definitions*; for details about the filenames for the sample UOC routines, see *8.1.7 Sample import information editing UOC routine*, *8.2.6 Sample column data editing UOC routine*, and *8.3.6 Sample send data UOC routine*.

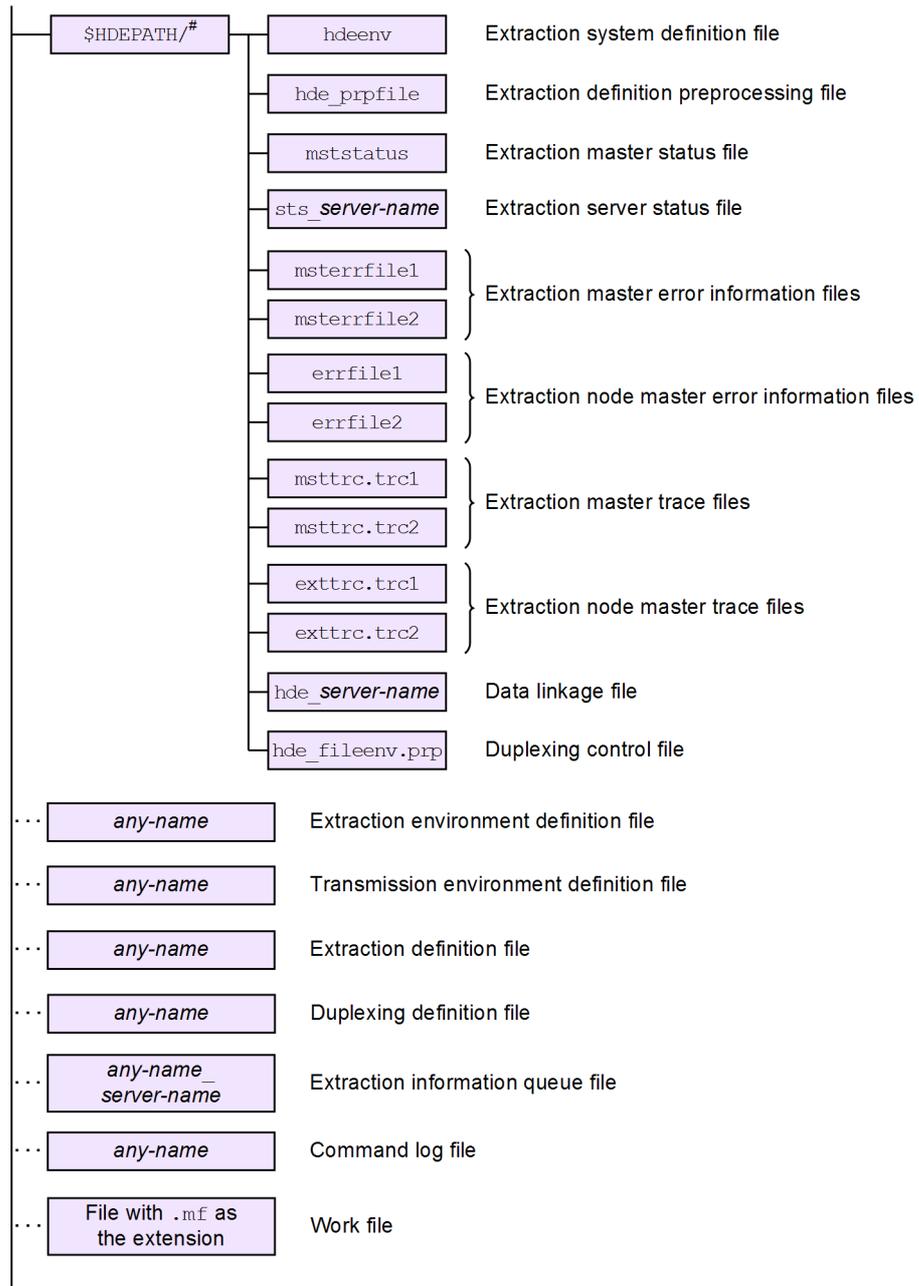
#2: This file is applicable when JP1/Cm2 is used to operate and manage Datareplicator.

For details about how to use JP1/Cm2 for operations management, see *3.4 Using JP1/Cm2 for operations management*.

2.3.2 Directory structure of a source Datareplicator

Figure 2-2 shows a source Datareplicator's directory structure, and Table 2-6 lists the file contents.

Figure 2-2: Source Datareplicator's directory structure (UNIX)



#

Directory that stores the Datareplicator definitions that are created in the source system.

Table 2-6: Contents of directories and files created by a source Datareplicator (UNIX)

Creation time	Directory and file name ^{#1}	Description
Created by the user	<code>\$HDEPATH/hdeenv</code>	Extraction system definition file
	<i>any-directory/any-name</i>	Extraction environment definition file
	<i>any-directory/any-name</i>	Transmission environment definition files
	<i>any-directory/any-name</i>	Extraction definition file
	<i>any-directory/any-name</i>	Duplexing definition file
Source Datareplicator's initial startup	<code>\$HDEPATH/hde_prpfile</code>	Extraction definition preprocessing file
	<i>any-directory/any-name_server-name</i>	Extraction information queue files
	<code>\$HDEPATH/mststatus</code>	Extraction master status file
	<code>\$HDEPATH/sts_server-name</code>	Extraction server status file
	<code>\$HDEPATH/msterrfile1</code> <code>\$HDEPATH/msterrfile2</code>	Extraction master error information files
	<code>\$HDEPATH/errfile1^{#2}</code> <code>\$HDEPATH/errfile2</code>	Extraction node master error information files
	<code>\$HDEPATH/msttrc.trc1</code> <code>\$HDEPATH/msttrc.trc2</code>	Extraction master trace files
	<code>\$HDEPATH/exttrc.trc1^{#3}</code> <code>\$HDEPATH/exttrc.trc2</code>	Extraction node master trace files
	<code>\$HDEPATH/hde_server-name</code>	Data linkage file
	<i>any-directory/any-name^{#4}</i>	Command log files
	<code>\$HDEPATH/hde_fileenv.prp</code>	Duplexing control file
	<i>any-directory/file-with-extension-.mf^{#5}</i>	Work file

^{#1}: The HDEPATH environment variable specifies the directory used to create the source Datareplicator's definitions.

#2: If you specify `true` in the `errfile_unique` extraction system definition operand, `_host-name` is attached to the filename (the resulting filenames become `errfile1_host-name` and `errfile2_host-name`).

#3: If you specify `true` in the `errfile_unique` extraction system definition operand, `_host-name` is attached to the filename (the resulting filenames become `exttrc_host-name.trc1` and `exttrc_host-name.trc2`).

#4: Specify the absolute or relative path name of the command log file in the `hde_command_log_file` environment variable. If the relative path name is specified, the system assumes `$HDEPATH/relative-path-name` as the absolute path name. If you omit the `hde_command_log_file` environment variable, the system assumes `$HDEPATH/hdecmdlog` as the name of the command log file.

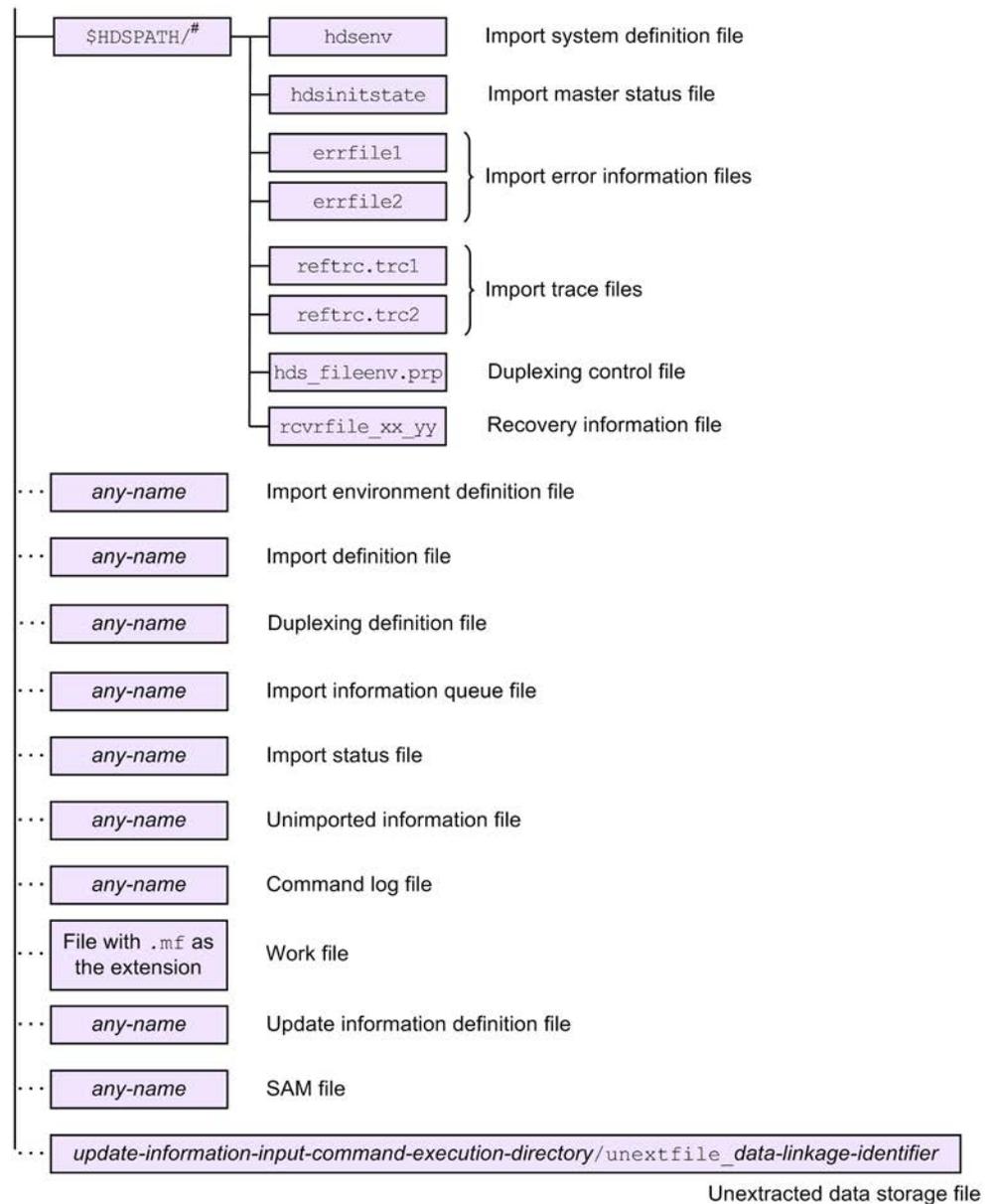
The actual command log file name has 1 or 2 appended at the end of the specified name. Make sure that the length of the actual command log file name does not exceed the *maximum length of OS path - 1*.

#5: This is a work file that is created automatically by Datareplicator when dual files are used. Do not delete this file while Datareplicator is running.

2.3.3 Directory structure of a target Datareplicator

Figure 2-3 shows a target Datareplicator's directory structure, and Table 2-7 lists the file contents.

Figure 2-3: Target Datareplicator's directory structure (UNIX)



#

Directory that stores the Datareplicator definitions that are created in the target system.

Table 2-7: Contents of directories and files created by a target Datareplicator (UNIX)

Creation time	Directory and filename^{#1}	Description
Created by the user	\$HDSPATH/hdsenv	Import system definition file
	<i>any-directory/any-name</i>	Import environment definition files
	<i>any-directory/any-name</i>	Import definition files
	<i>any-directory/any-name</i>	Duplexing definition file
While connected with the source system	\$HDSPATH/rcvrfile_xx_yy ^{#2}	Recovery information file
Target	<i>any-directory/any-name</i>	Import information queue files
Target Datareplicator's initial startup	<i>any-name</i>	Import status files
	\$HDSPATH/hdsinitstate	Import master status file
	\$HDSPATH/errfile1 \$HDSPATH/errfile2	Import error information files
	\$HDSPATH/reftrc.trc1 \$HDSPATH/reftrc.trc2	Import trace files
	<i>any-directory/any-name</i>	Unimported information files
	<i>any-directory/any-name^{#3}</i>	Command log files
	\$HDSPATH/hds_fileenv.prp	Duplexing control file
	<i>any-directory/file-with-extension-.mf^{#4}</i>	Work file
Created by the user	<i>any-directory/any-name</i>	Update information definition file ^{#5}
Transferred by the user	<i>any-directory/any-name</i>	SAM file ^{#6}
During execution of the update information input command	<i>update-information-input-command-execution-directory/unextfile_data-linkage-identifier</i>	Unextracted data storage file ^{#7}

#1: The HDSPATH environment variable specifies the directory used to create the target

Datareplicator's definitions.

#2: *xx* indicates the target Datareplicator identifier; *yy* indicates the data linkage identifier.

#3: Specify the absolute or relative path name of the command log file in the `hds_command_log_file` environment variable. If the relative path name is specified, the system assumes `$HDSPATH/relative-path-name` as the absolute path name. If you omit the `hds_command_log_file` environment variable, the system assumes `$HDSPATH/hdscmdlog` as the name of the command log file.

The actual command log file name has 1 or 2 appended at the end of the specified name. Make sure that the length of the actual command log file name does not exceed the *maximum length of OS path - 1*.

#4: This is a work file that is created automatically by Datareplicator when dual files are used. Do not delete this file while Datareplicator is running.

#5: Of mainframe databases when a SAM file is used, you must create an update information definition file for a target Datareplicator that extracts PDM2 E2 data. There is no need to create this file for any other Datareplicator.

#6: The update information extraction SAM file for the mainframe is sent by the file transfer program. You must create a SAM file when the target Datareplicator is linked to a mainframe database that uses SAM files (PDM2 E2 or RDB1 E2). There is no need to create this file for any other Datareplicator.

#7: The unextracted data storage file is created/re-created each time an update information input command is executed. Its filename is `unextfile_` plus the data linkage identifier specified during execution of the update information input command.

2.4 Specifying environment variables (UNIX)

Once you have installed the required products, you must specify *environment variables* before you can execute your Datareplicator. This section explains the environment variables that you need to specify for a source Datareplicator and for a target Datareplicator.

2.4.1 Environment variables for a source Datareplicator

This section explains the environment variables for a source Datareplicator and includes specification examples.

(1) Information specified in the environment variables for a source Datareplicator

You specify the environment variables for a source Datareplicator in the user environment where the source Datareplicator's commands are executed. The following table shows the information that is specified in the environment variables for a source Datareplicator.

Table 2-8: Information specified in the environment variables for a source Datareplicator

Environment variable	Description
PATH	Specify the name of the source Datareplicator's commands library. ^{#1}
LANG	Specify the character code set to be used in the source Datareplicator's messages and definitions. Be sure to specify the correct character code set based on the information specified in the <code>dblocale</code> and <code>msglocale</code> operands in the extraction system definition.
TZ	Specify the time zone of the messages output from the source Datareplicator.
PDDIR ^{#2, #3}	Specify the source HiRDB directory.
PDCONFPATH ^{#2, #3}	Specify the directory for storing the source HiRDB's system definition files.
PDNAMEPORT ^{#2}	Specify the source HiRDB's port number.
PDHOST ^{#2}	Specify the source HiRDB's host name.
PDUSER ^{#2}	Specify the user connected to the source HiRDB.
PDLANG ^{#2}	If the character code set used in the source HiRDB is UTF-8, specify UTF-8. If the character code set used in the source HiRDB is not UTF-8, there is no need to specify this environment variable.
SHLIB_PATH ^{#2, #3, #4, #5}	Specify <code>\$PDDIR/lib</code> . If you use the Linux (IPF) or HP-UX (IPF) edition, specify <code>\$PDDIR/lib:\$PDDIR/client/lib</code> .

Environment variable	Description
HDEPATH	<p>Specify the source Datareplicator's operation directory. The source Datareplicator's definition files, status files, error information files, and activity trace files are created in this directory.</p> <p><i>Notes:</i></p> <ul style="list-style-type: none"> • Make sure that the specified directory path name is 64 bytes or less. • If the source and target Datareplicators are located on the same machine, do not specify the same directory in the HDEPATH and HDSPATH environment variables. • If the source HiRDB is a parallel server consisting of multiple server machines, specify the same operation directory for the source Datareplicator at each server machine.
HDE_BIN_COL_MAXLEN	<p>Specify a value (kilobytes) that is smaller than the actual definition length of any BLOB-type column that is handled internally in Datareplicator. If a BLOB-type column has a definition length of 2 GB or greater but its actual data is small, this environment variable enables the column data to be linked without having to redefine the table.</p> <p><i>Notes:</i></p> <ul style="list-style-type: none"> • The definition length of each BLOB-type column must be 262,143 KB or less. • The value specified in this environment variable applies to all BLOB-type columns that are subject to extraction. • For details about the handling of BLOB-type columns whose definition length exceeds the value specified in this environment variable, see 6.4.3(4) <i>Notes about the HDE_BIN_COL_MAXLEN environment variable.</i>
EXTSHM (AIX only)	<p>Specify ON, which indicates that there is no limit to the number of shared memory spaces in the process space.</p> <p>Specify it if a shortage occurs on the shared memory. If there is no shortage, specification of this environment variable is optional.</p>
PSALLOC (AIX only)	<p>Specify early, which indicates that a required paging space is to be allocated immediately during memory allocation. With AIX, paging space is not normally allocated during memory allocation.</p> <p>Make sure that this variable is set.</p>
LDR_CNTRL (AIX only)	<p>MAXDATA=<i>memory-size</i></p> <p>Specify the size of memory required for program execution. Available memory size is 2 gigabytes in the range from 0x20000000 to 0x80000000. Set the memory in units of 256 megabytes. The memory must be allocated in good balance because the memory size is common to both shared memory and mmap memory.</p> <p>Specify it if a shortage occurs on the shared memory. If there is no shortage, specification of this environment variable is optional. If a memory shortage occurs, set 0x20000000 and then start. If a memory shortage recurs, increase the memory in increments of 0x10000000 and then restart.</p>
NODISCLAIM (AIX only)	<p>For free(), set True as the call processing method, indicating that issuance of nodisclaim() is to be suppressed.</p> <p>Make sure that this variable is set.</p>

#1: The name of the commands library is `opt/hirdbds/bin/`.

#2: This is a source HiRDB environment variable. For details about the HiRDB design, see the *HiRDB Version 9 Installation and Design Guide*.

#3: If the source HiRDB is a parallel server and this setting varies from one server to another, extraction system definitions can be used to define this information for each server machine. If definitions for a specific server are omitted from the extraction system definitions, the corresponding server uses the values of the environment variables that were specified for the user environment where the source Datareplicator's commands are executed.

#4: This environment variable for the Solaris and Linux Datareplicators is `LD_LIBRARY_PATH`. The environment variable for the AIX Datareplicator is `LIBPATH`.

#5: If the value specified in this environment variable exceeds 255 bytes in length, an invalid value error results at the source Datareplicator. If it is necessary to specify a value greater than 255 bytes in length, specify in the `node_shlibpath` operand of the extraction system definition only as much of the library path required for the source Datareplicator as can be specified in no more than 255 bytes. By specifying the `node_shlibpath` operand, you can avoid having an invalid value error for the environment variable.

To apply the `node_shlibpath` operand value, you must initialize the source Datareplicator. If you expect the length of the environment variable value might exceed 255 bytes in the future, we recommend that you specify the `node_shlibpath` operand during initial configuration.

(2) Examples of source Datareplicator environment variables

The following are examples of environment variables for a source Datareplicator. These examples are for the HP-UX Datareplicator.

Bourne shell (sh)

```
$ PATH=$PATH:/opt/hirdbds/bin
$ LANG=ja_JP.SJIS#
$ HDEPATH=/opt/hirdbds/define
$ export PATH LANG HDEPATH
```

C shell (csh)

```
% set path=($path /opt/hirdbds/bin)
% setenv LANG ja_JP.SJIS#
% setenv HDEPATH /opt/hirdbds/define
```

#: For details about how to specify the `LANG` environment variable, see the applicable manual.

These examples assume that the source Datareplicator definitions are stored in the /

`opt/hirdb/define/` directory. These examples omit the TZ environment variable as well as the HiRDB environment variables.

For details about how to specify the TZ environment variable, see the applicable manual.

For an example of specifying a source HiRDB's environment variables (\$PDDIR, \$PDCONFPATH, \$PDNAMEPORT, \$PDHOST, \$PDUSER, and \$SHLIB_PATH), see the *HiRDB Version 9 Installation and Design Guide*.

2.4.2 Environment variables for a target Datareplicator

This section explains the environment variables for a target Datareplicator and includes specification examples.

(1) Information specified in the environment variables for a target Datareplicator

You specify the environment variables for a target Datareplicator in the user environment where the target Datareplicator's commands are executed. The following table shows the information that is specified in the environment variables for a target Datareplicator.

Table 2-9: Information specified in the environment variables for a target Datareplicator

Environment variable	Description
PATH	Specify the name of the target Datareplicator's commands library. ^{#1}
LANG	Specify the character code set to be used in the target Datareplicator's messages and definitions. Be sure to specify the correct character code set based on the information specified in the <code>dblocale</code> and <code>msglocale</code> operands in the import system definition.
TZ	Specify the time zone of the messages output from the target Datareplicator.
PDDIR ^{#2}	Specify the target HiRDB directory.
PDNAMEPORT ^{#2}	Specify the target HiRDB's port number.
PDHOST ^{#2}	Specify the target HiRDB's host name.
PDLANG ^{#2}	If the character code set used in the target HiRDB is UTF-8, specify UTF-8. If the character code set used in the target HiRDB is not UTF-8, there is no need to specify this environment variable.
SHLIB_PATH ^{#2, #3}	Specify <code>\$PDDIR/lib</code> . If you use the Linux (IPF) or HP-UX (IPF) edition, specify <code>\$PDDIR/lib:\$PDDIR/client/lib</code> .

2. Environment Setup

Environment variable	Description
HDS_PATH	<p>Specify the target Datareplicator's operation directory. The target Datareplicator's definition files, status files, error information files, and activity trace files are created in this directory.</p> <p><i>Notes:</i></p> <ul style="list-style-type: none"> • Make sure that the specified directory path name is 64 bytes or less. • If the source and target Datareplicators are located on the same machine, do not specify the same directory in the HDEPATH and HDS_PATH environment variables.
HDS_MST_STDCLOSE	<p>Specify whether the standard input, standard output, and standard error output are to be closed by the target Datareplicator.</p> <p>When this environment variable is omitted or TRUE is specified: The standard input, standard output, and standard error output will be closed by the target Datareplicator.</p> <p>When FALSE is specified: The standard input, standard output, and standard error output will not be closed by the target Datareplicator.</p> <p>When you specify FALSE, an import information editing UOC routine can use the standard input, standard output, and standard error output.</p> <p>If an import information editing UOC routine uses the standard input, standard output, or standard error output when this environment variable is not specified, data might not be imported correctly. If you plan to have an import information editing UOC routine use the standard input, standard output, or standard error output, make sure that you specify FALSE in this environment variable.</p>
HDS_RFI_ELANG ^{#4}	<p>Specify the character encoding used in the source system.</p> <p>If this information has not been specified, the system automatically identifies the character encoding on the basis of the import status file. This automatic identification is not available when the definition information is displayed immediately after the target system has been initialized. If automatic identification is not available, ja_JP.SJIS is assumed.</p>
HDS_RFI_PLANG ^{#4}	<p>Specify the character encoding used in the target system.</p> <p>If this information has not been specified, ja_JP.SJIS is assumed.</p>

Environment variable	Description
HDSCLTWAITTIME	<p>Specify whether the values of PDCWAITTIME (client's maximum wait time) and PDSWAITTIME (server's maximum wait time)^{#5} specified in the HiRDB client environment definition are to take effect on the target Datareplicator.</p> <p>This environment variable is optional; we recommend that you omit it if it is not needed.</p> <p>USER Uses the values specified in the HiRDB client environment definition.</p> <p>REPL Ignores the values specified in the HiRDB client environment definition and sets the following values:</p> <ul style="list-style-type: none"> • PDCWAITTIME Sets 0 (wait until there is a response). • PDSWAITTIME Sets the commit_wait_time operand value in the import system definition^{#6} + 600 (seconds). If the import transaction synchronization facility is used, the syncwait_limit_tim operand value in the import system definition + 600 (seconds) is set. If the specified setting exceeds 65,535, 65535 is set. <p>When this environment variable is omitted: If an import information editing UOC routine is used, USER is assumed. If the import SQL process is used, REPL is assumed.</p> <p>When USER is specified: Set in PDCWAITTIME and PDSWAITTIME in the client environment definition values that are greater than the following operand values:</p> <ul style="list-style-type: none"> • PDCWAITTIME pd_lck_wait_timeout operand in HiRDB • PDSWAITTIME commit_wait_time and syncwait_limit_time operands in the import system definition <p>If the values set in PDCWAITTIME and PDSWAITTIME are too small, HiRDB's monitoring interval might be reached. In such a case, the target Datareplicator's SQL process detects an SQL error, resulting in abnormal termination.</p>
EXTSHM (AIX only)	<p>Specify ON, which indicates that there is no limit to the number of shared memory spaces in the process space.</p> <p>Specify it if a shortage occurs in the shared memory. If there is no shortage, specification of this environment variable is optional.</p>
PSALLOC (AIX only)	<p>Specify early, which indicates that a required paging space is to be allocated immediately during memory allocation. With AIX, paging space is not normally allocated during memory allocation.</p> <p>Make sure that this variable is set.</p>

2. Environment Setup

Environment variable	Description
LDR_CNTRL (AIX only)	<p><i>MAXDATA=memory-size</i></p> <p>Specify the size of the memory required for program execution. The maximum available memory size is 2 gigabytes, in the range from 0x20000000 to 0x80000000. Set the memory in units of 256 megabytes. The memory must be allocated in good balance because the memory size is common to both shared memory and mmap memory.</p> <p>Specify it if a shortage occurs in the shared memory. If there is no shortage, specification of this environment variable is optional. If a memory shortage occurs, set 0x20000000 and then start. If the memory shortage recurs, increase the memory in increments of 0x10000000 and then restart.</p>
NODISCLAIM (AIX only)	<p>For <code>free()</code>, set <code>True</code> as the call processing method, indicating that issuance of <code>nodisclaim()</code> is to be suppressed.</p> <p>Make sure that this variable is set.</p>

#1

The name of the commands library is `opt/hirdbds/bin/`.

#2

This is a target HiRDB environment variable. For details about the HiRDB design, see the *HiRDB Version 9 Installation and Design Guide*.

#3

This environment variable for the Solaris and Linux Datareplicators is `LD_LIBRARY_PATH`. This environment variable for the AIX Datareplicator is `LIBPATH`.

#4

The following table shows the values of `HDS_RFI_ELANG` and `HDS_RFI_PLANG`:

Environment variable	Character encoding			
	JIS8 Shift JIS	EUC	Unicode (UTF-8)	EBCDIC/KEIS EBCDIK/KEIS
<code>HDS_RFI_ELANG</code>	<code>ja_JP.SJIS</code>	<code>ja_JP.UJIS</code>	<code>ja_JP.UTF8</code>	EBCDIK
<code>HDS_RFI_PLANG</code>	<code>ja_JP.SJIS</code>	<code>ja_JP.UJIS</code>	<code>ja_JP.UTF8</code>	--

Legend:

--: Not applicable

#5

This does not apply to PDSWATCHTIME in the HiRDB client environment definition. The value specified in the HiRDB client environment definition always takes effect.

#6

If the `commit_wait_time` operand is specified in the import environment definition, the value specified in the import environment definition takes effect.

(2) Examples of target Datareplicator's environment variables

The following are examples of environment variables for a target Datareplicator. These examples are for the HP-UX Datareplicator.

Bourne shell (sh)

```
$ PATH=$PATH:/opt/hirdbds/bin
$ LANG=ja_JP.SJIS#
$ HDSPATH=/opt/hirdbds/define
$ HDSCLTWAITTIME=USER
$ export PATH LANG HDSPATH HDSCLTWAITTIME
```

C shell (csh)

```
% set path=( $path /opt/hirdbds/bin )
% setenv LANG ja_JP.SJIS#
% setenv HDSPATH /opt/hirdbds/define
% setenv HDSCLTWAITTIME USER
```

#: For details about how to specify the LANG environment variable, see the applicable manual.

These examples assume that the target Datareplicator definitions are stored in the `/opt/hirdb/define/` directory. These examples omit the TZ environment variable as well as the HiRDB environment variables.

For details about how to specify the TZ environment variable, see the applicable manual.

For an example of specifying a target HiRDB's environment variables (`$PDDIR`, `$PDCONFPATH`, `$PDNAMEPORT`, `$PDHOST`, and `$SHLIB_PATH`), see the *HiRDB Version 9 Installation and Design Guide*.

2.5 Setting up the communications environment (UNIX)

Before you can execute Datareplicator, you must set up the communications environment. This section describes the settings for the communications environment.

2.5.1 Setting up a source Datareplicator's communications environment

This section explains the settings for a source Datareplicator's communications environment and includes examples.

(1) Settings for a source Datareplicator's communications environment

The following shows the settings for a source Datareplicator's communications environment.

Table 2-10: Settings for a source Datareplicator's communications environment

Communications environment	Settings	Server machine subject to setup	
		Source HiRDB is a single server	Source HiRDB is a parallel server
/etc/services	Specify the <i>service-name</i> and <i>port-number</i> ^{#1} to be used for communication with the target system. Specify this service name in the <i>hdeservice</i> operand in the transmission environment definition. Use the same port number as for the target system.	SDS	BES
/etc/services	Specify the <i>service-name</i> and <i>port-number</i> ^{#1} to be used for communication with the extraction master process and the extraction node master process. Specify this service name in the <i>mstservice</i> operand of the extraction system definition. Use the same port number for the server containing the system manager and back-end server.	SDS	MGR, BES ^{#2}
/etc/hosts	Specify the <i>IP-address</i> and <i>host-name</i> to be used for communication with the target system. Specify this host name in the <i>hdehost</i> operand of the transmission environment definition.	SDS	BES
/etc/inetd.conf	Register the <i>entry</i> used to start the extraction node master process.	SDS	MGR, BES

SDS: Server machine at which a single server is defined for the source HiRDB.

MGR: Server machine at which a system manager is defined for the source HiRDB.

BES: Server machine at which a back-end server (including back-end servers that do not contain any database subject to data extraction) is defined for the source HiRDB.

#1: You cannot specify a port number that has already been registered in the `services` file or that is being used by other software.

#2: Set up the communications environment for the server machine at which the system manager is defined with the source HiRDB and for all server machines at which a back-end server is defined.

(2) Examples of a communications environment for a source Datareplicator

The following are examples of setting up a source Datareplicator's communications environment.

(a) Example of `/etc/services` for a source Datareplicator

The following is an example of specifying the service name and port number in `/etc/services`:

```
service-name port-number/tcp
```

Note:

The underlined portion is a fixed value.

service-name

Specify any name.

port-number

Specify a valid value in the communications environment. No other service can be using this port number.

(b) Example of `/etc/hosts` for a source Datareplicator

The following is an example of specifying the IP address and host name in `/etc/hosts`:

```
IP-address host-name
```

IP-address

Specify any address.

host-name

Specify any name.

(c) Example of `/etc/inetd.conf` for a source Datareplicator

The following is an example of registering an entry to start the extraction node master

process in `/etc/inetd.conf`.

For HP-UX or Solaris Datareplicator

```
hdeserv stream tcp nowait user1 /opt/hirdbds/lbin/hdenodemst  
hdenodemst
```

Note:

The underlined portions are fixed values. These examples are for the HP-UX Datareplicator.

To apply the change to the `/etc/inetd.conf` settings, execute the following command:

- `kill -HUP inetd-process-ID`

For AIX Datareplicator

```
hdeserv stream tcp nowait user1 /bin/env env [EXTSHM=ON]  
PSALLOC=early  
NODISCLAIM=true  
[LDR_CNTRL=MAXDATA=0x ... ] /opt/hirdbds/lbin/hdenodemst
```

Note:

The underlined values are fixed.

To apply the change to the `/etc/inetd.conf` settings, execute the following command:

- `kill -HUP inetd-process-ID`

For Linux Datareplicator

In the Linux edition, replace `/etc/inetd.conf` with `/etc/xinetd.conf` and replace the `inetd` process with the `xinetd` process.

The following shows an example of `/etc/xinetd.conf`. Modify the underlined portion as appropriate to your environment.

```
service hdeserv  
{  
    socket_type      = stream  
    protocol        = tcp  
    wait            = no  
    user            = user1  
    server          = /opt/hirdbds/lbin/hdenodemst  
}
```

Note:

To apply the change to the `/etc/xinetd.conf` settings, execute the

following command:

```
/sbin/service xinetd reload
```

hdeserv

Specify the service name that is specified in `mstservice` in the extraction system definition.

user1

Specify a user name that belongs to the same group as the source HiRDB user.

2.5.2 Setting up a target Datareplicator's communications environment

This section explains the settings for a target Datareplicator's communications environment and includes examples. You set up a target Datareplicator's communication environment for the service machine used to execute the target Datareplicator's commands.

(1) Settings for a target Datareplicator's communications environment

The following table shows the settings for a target Datareplicator's communications environment.

Table 2-11: Settings for a target Datareplicator's communications environment

Communications environment	Settings
<code>/etc/services</code>	Specify the <i>service-name</i> and <i>port-number</i> to be used for communication with the source system. Specify this service name in the <code>hdsservice</code> operand in the import system definition. Use the same port number as for the source system.
XNF network definition	If the OSI protocol (channel connection) is used for communication [#] , specify the <i>UCE name</i> and <i>T selector value</i> to be used for communication with the source system. Specify this T selector value in the <code>reflect_tselector</code> operand of the import system definition. Use the same T selector value as for the source system.

[#]: The OSI protocol (channel connection) is supported only when the operating system being used is HP-UX; in the case of AIX, Linux, or Windows, only TCP/IP is supported as the communications protocol.

(2) Example of a communications environment for a target Datareplicator

The following is an example of setting up a target Datareplicator's communications environment. For details about the TCP/IP network environment settings, see the *HI-UX/WE2 Hitachi CSMA/CD Network CD105 (TCP/IP)* manual. For details about the XNF network definitions, see the *XNF/S-E2 Definition* manual.

(a) Example of /etc/services for a target Datareplicator

The following is an example of specifying the service name and port number in /etc/services:

service-name *port-number*/tcp

Note

The underline portion is a fixed value.

service-name

Specify any name.

port-number

Specify a valid value in the communications environment. No other service can be using this port number.

2.6 Installing Datareplicator (Windows)

This section explains the procedure for installing Windows Datareplicator.

2.6.1 Preparations before installation

Before installing Datareplicator, check the following:

- Operating environment
- User privileges

(1) *Checking the operating environment*

Checking the operating environment involves the following:

- Check that the machine on which Datareplicator is to be installed has sufficient free space.

For details about the disk space required for Datareplicator, see *4.6.8 Designing the source Datareplicator's resources* or *4.7.7 Designing the target Datareplicator's resources*.

- Check that all products required for Datareplicator operation have been installed.

For details about the products required for Datareplicator operation, see *2.1 Products associated with Datareplicator*.

(2) *Checking user privileges*

Check that the person installing Datareplicator is a user with Administrator privileges.

2.6.2 Server machines where Datareplicator is installed

The server machines where Datareplicator is installed depend on whether the HiRDB system being used is a HiRDB/Single Server or a HiRDB/Parallel Server. When a HiRDB/Parallel Server is being used, the server machines depend on whether it is the source or target system.

The following table shows the server machines where Datareplicator can be installed.

Table 2-12: Server machines where Datareplicator is installed

Type of HiRDB system	Source system	Target system
HiRDB/Single Server	Install on the server machine where the single server is located.	Install on the server machine where the single server is located.
HiRDB/Parallel Server	Install on all server machines that contain the following servers: <ul style="list-style-type: none"> • System manager (MGR) • Back-end servers (BES) 	Install one Datareplicator on each server machine where the HiRDB client facility is used.

2.6.3 Installing Datareplicator

See 2.1 *Products associated with Datareplicator* and install all products that are required in order to use Windows Datareplicator. You can install HiRDB before Datareplicator; or vice versa; however, you must install HiRDB on a server machine before you can specify the Datareplicator's environment variables on that server machine. Once you have installed Datareplicator, you must restart Windows.

If you specify the Datareplicator environment variables before you install Datareplicator, you can restart Windows only once. For details about how to specify the Datareplicator environment variables, see 2.8 *Specifying environment variables (Windows)*.

(1) Installation procedure

The procedure for installing Datareplicator from the installation integrated CD-ROM is as follows. Before you start the installation procedure, you must exit Datareplicator and all Windows applications.

To install Datareplicator:

1. Insert the HiRDB integrated CD-ROM disk into the disk drive, and run `hcd_inst.exe`.

The Hitachi integrated installer starts.

2. Click the **Next** button according to the instruction displayed in the window.
3. Enter your name and company name, and then click the **Next** button.

To use the displayed name and company name, there is no need to enter this information

4. Specify the installation directory, and then click the **Next** button. The default installation directory is `Windows-installation-target-drive:\Program Files\HITACHI\hirdbds`.

To install in some other directory, click the **Browse** button and specify the desired installation directory.^{#1} If the specified directory is not found, the system issues a message asking whether you want it to create the directory.

5. Select the setup method, and then click the **Next** button.

If you select **Standard**, the target Datareplicator, the source Datareplicator, and operations management^{#2} are installed.

If you select **Compact**, the target Datareplicator and the source Datareplicator are installed.

If you select **Custom**, you can select the components to be installed.

6. When **HiRDB Datareplicator** is displayed as the name of the program folder or

group to be registered, click the **Next** button.

To register a group with a different name, specify the appropriate name. To register an existing group, select the group.

7. When the current settings are displayed, click the **Next** button to start copying files.

Installation of Datareplicator begins.

8. When installation of Datareplicator is completed, restart Windows.

#1

You can specify only a local drive. Do not specify a network drive.

#2

If you attempt to install operations management but it has already been installed, an error might occur during file copying. In such a case, terminate the SNMP service, and then reinstall operations management.

2.6.4 Information registered during installation

This section describes the information that is registered during installation.

(1) Services

The following table shows the types of services that are registered during installation, the names displayed when the **Services** icon in **Control Panel** is opened, and the startup settings.

Table 2-13: Services registered during installation

Type	Displayed name	Type of startup [#]	Log on [#]
Extraction service	HiRDB Datareplicator (Source Site)	Manual	System account
Node master process startup service	HiRDB Datareplicator (Source Site NMT)	Manual	System account
Import service	HiRDB Datareplicator (Target Site)	Manual	System account
Status monitoring service	HiRDB Datareplicator (Agent)	Manual	System account

[#]: To change after installation, go to **Control Panel** and double-click the **Services** icon. You can change the account to a user account with `Administrators` privilege in addition to the system account.

(2) Icon

An icon for the first file to be read

(*Datareplicator-installation-directory\Readme.txt*) is registered in the **HiRDB Datareplicator** folder or group.

There is no icon for an uninstallation program, because you choose the **Add/Remove Programs** icon in **Control Panel** in order to uninstall Datareplicator.

(3) System environment variable

The name of the Datareplicator's commands library is added to the `PATH` environment variable. The following is the name of the commands library:

Datareplicator-installation-directory\bin

2.6.5 Post-installation procedure

In the case of a source Datareplicator, you must perform the following after installation is completed:

- Specify the name (`hdenmserv`) of the node master process startup service in *Windows-system-directory\drivers\etc\services* (referred to hereafter as the `services` file). For details about the specification and format, see 2.9 *Setting up the communications environment (Windows)*.
- Set up the node master process startup service so that it will start automatically.

To set up the node master process startup service:

1. In **Control Panel**, double-click the **Services** icon.
2. Under **Service**, choose **HiRDB Datareplicator (Source Site NMT)**.
3. Click the **Startup** button.
4. In the Service dialog box, select **Automatic** in the **Startup Type** area.
5. Click the **OK** button.

Notes

You must set up the `services` file before exiting or restarting Windows. Otherwise, the node master process startup service results in an error when Windows starts.

If the source HiRDB is a HiRDB/Parallel Server, it is necessary for the node master startup service to start automatically at all HiRDB nodes where the source Datareplicator is used.

2.6.6 Uninstalling Datareplicator

This section explains the procedure for uninstalling Datareplicator. Before you uninstall Datareplicator, you must exit all active HiRDB Datareplicator services.

To uninstall Datareplicator:

1. In **Control Panel**, double-click the **Add/Remove Programs** icon.

2. Choose the **Install/Uninstall** tab.
3. From the list of installed applications, select **HiRDB Datareplicator**, and then click the **Add/Remove** button.
4. Click the **OK** button according to the installations displayed on the screen.
5. When uninstallation is complete, click the **OK** button.

2.7 Directory structure (Windows)

The files used by Windows Datareplicator include files created automatically during installation, definition files used to set up the system environment, and files created automatically during Datareplicator's initial startup.

This section explains the Windows Datareplicator's directory structure.

2.7.1 Directories created when Datareplicator is installed

Figure 2-4 shows the directories that are created when Windows Datareplicator is installed, and Table 2-14 shows the file contents. These directories are the same for both the source and target Datareplicators.

Figure 2-4: Directories and files associated with Datareplicator

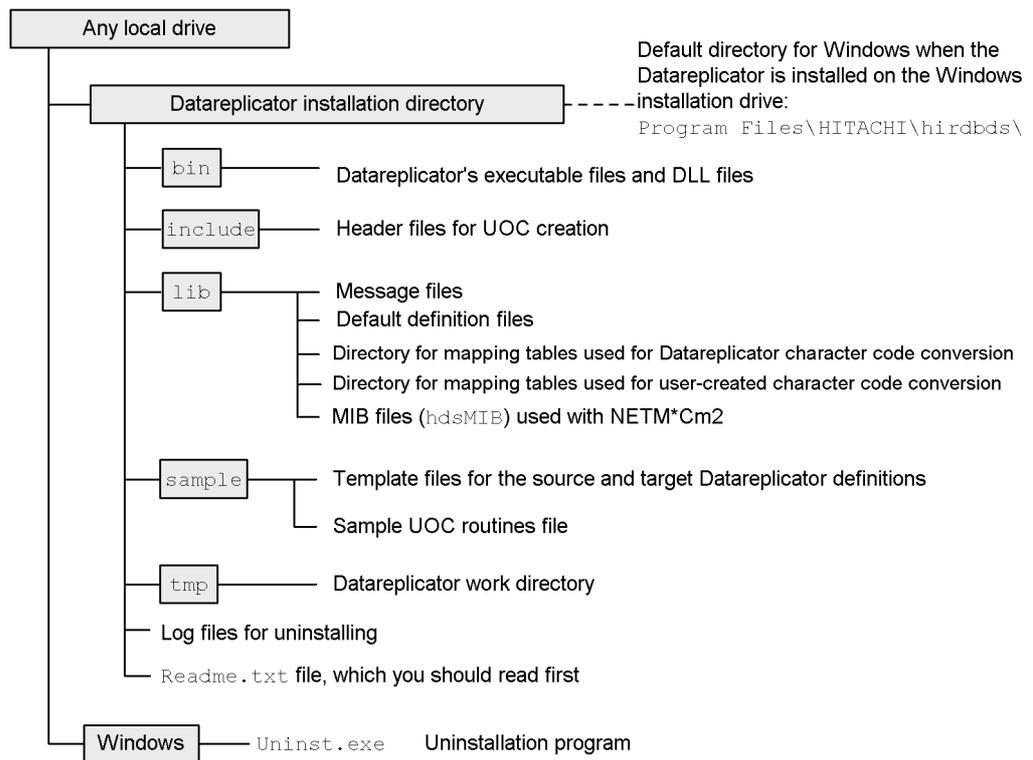


Table 2-14: Contents of directories and files created when Datareplicator is installed

Creation time	Directory and filename	Description
Installation	\bin\	Directory containing the Datareplicator's commands
	\include\	Header files for UOC creation
	\lib\	Datareplicator's shared libraries and message files
	\lib\maptable\	Directory for mapping tables used for Datareplicator character code conversion
	\lib\usermap\	Directory for mapping tables used for user-created character code conversion
	\lib\hdsMIB	MIB file used with JP1/Cm2 ^{#1}
	\sample\	Sample Datareplicator definition and UOC ^{#2}
	\tmp\	Datareplicator's work directory

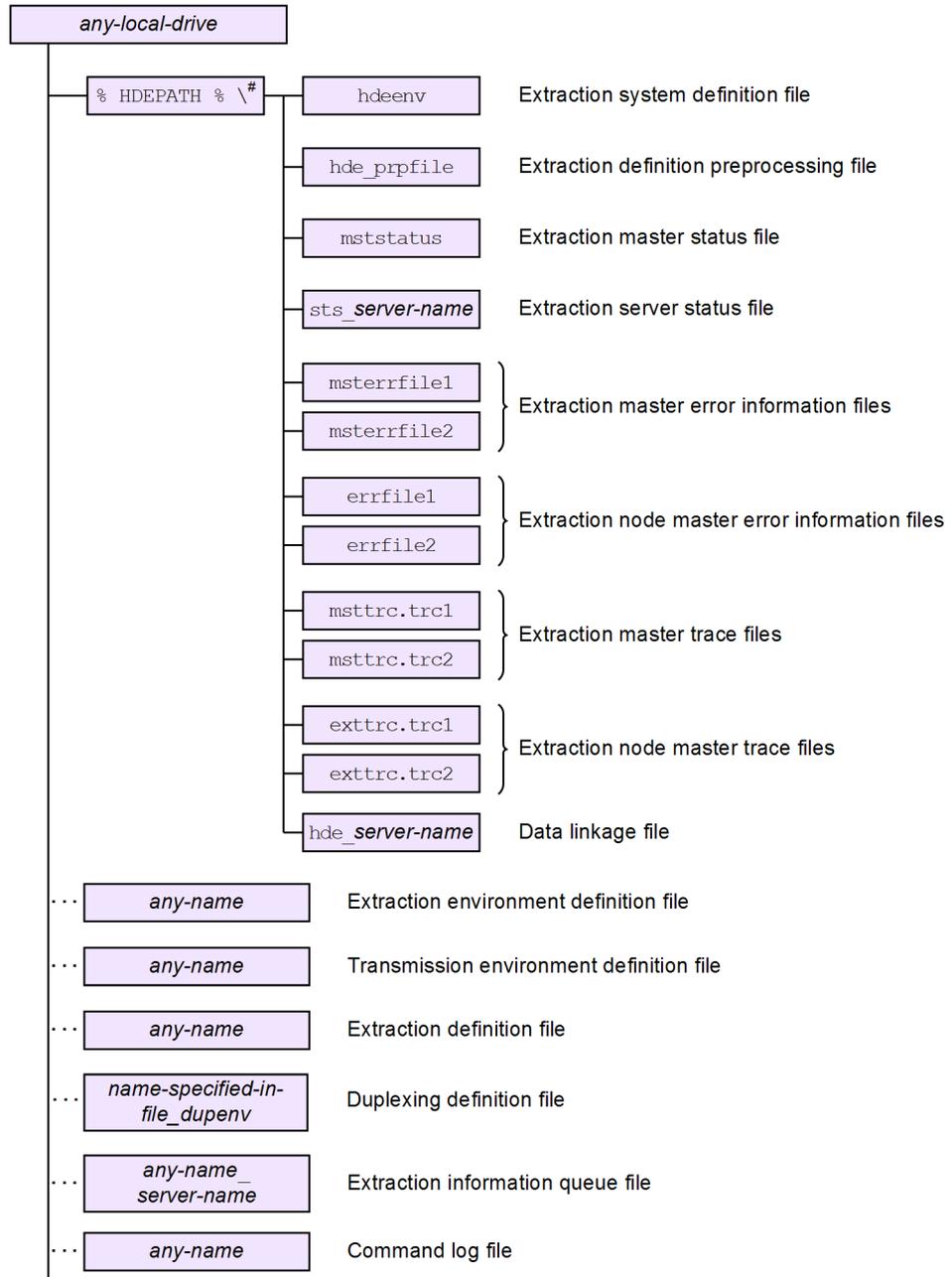
#1: This file is applicable when JP1/Cm2 is used to operate and manage Datareplicator. For details about how to use JP1/Cm2 for operations management, see *3.4 Using JP1/Cm2 for operations management*.

#2: For details about the filenames for the Datareplicator definition template, see *5.13.2 Examples of source Datareplicator definitions* or *5.13.3 Examples of target Datareplicator definitions*; for details about the filenames for the sample UOC routines, see *8.1.7 Sample import information editing UOC routine*, *8.2.6 Sample column data editing UOC routine*, and *8.3.6 Sample send data UOC routine*.

2.7.2 Directory structure of a source Datareplicator

Figure 2-5 shows a source Datareplicator's directory structure, and Table 2-15 lists the file contents.

Figure 2-5: Source Datareplicator's directory structure (Windows)



#

Environment variable that indicates the directory that stores the Datareplicator definitions.

Table 2-15: Contents of directories and files for a source Datareplicator (Windows)

Creation time	Directory and filename ^{#1, #2}	Description
Created by the user	%HDEPATH%\hdeenv	Extraction system definition file
	<i>any-name</i>	Extraction environment definition file
	<i>any-name</i>	Transmission environment definition files
	<i>any-name</i>	Extraction definition file
	<i>file-name-specified-in-file_dupenv-operand-in-extraction-system-definition-file</i>	Duplexing definition file
Source Datareplicator's initial startup	%HDEPATH%\hde_prpfile	Extraction definition preprocessing file
	<i>any-name_server-name</i>	Extraction information queue files
	%HDEPATH%\mststatus	Extraction master status file
	%HDEPATH%\sts_ <i>server-name</i>	Extraction server status file
	%HDEPATH%\msterrfile1 %HDEPATH%\msterrfile2	Extraction master error information files
	%HDEPATH%\errfile1 ^{#3} %HDEPATH%\errfile2	Extraction node master error information files
	%HDEPATH%\msttrc.trc1 %HDEPATH%\msttrc.trc2	Extraction master trace files
	%HDEPATH%\exttrc.trc1 ^{#4} %HDEPATH%\exttrc.trc2	Extraction node master trace files
	%HDEPATH%\hde_ <i>server-name</i>	Data linkage file
	<i>any-name</i> ^{#5}	Command log files

#1: Each directory is created under a local drive.

#2: The HDEPATH environment variable specifies the directory used to create the Datareplicator's definitions.

#3: If you specify `true` in the `errfile_unique` extraction system definition operand, `_host-name` is attached to the filename (the resulting filenames become `errfile1_host-name` and `errfile2_host-name`).

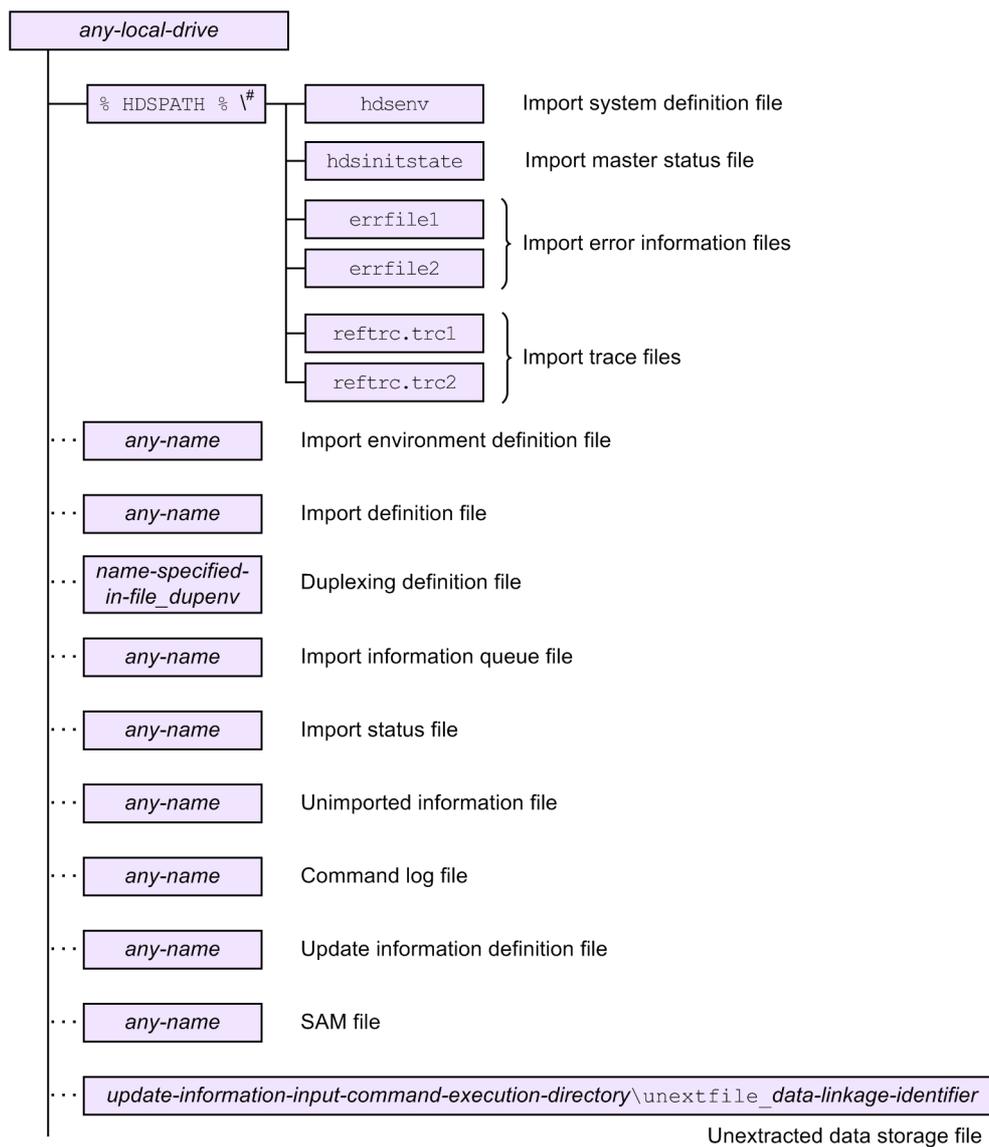
#4: If you specify `true` in the `errfile_unique` extraction system definition operand, `_host-name` is attached to the filename (the resulting filenames become `exttrc_host-name.trc1` and `exttrc_host-name.trc2`).

#5: Specify the absolute or relative path name of the command log file in the `hde_command_log_file` environment variable. If the relative path name is specified, the system assumes `%HDEPATH%\relative-path-name` as the absolute path name. If you omit the `hde_command_log_file` environment variable, the system assumes `%HDEPATH%\hdecmdlog` as the name of the command log file. The actual command log file name has 1 or 2 appended at the end of the specified name. Make sure that the length of the actual command log file name does not exceed the *maximum length of the OS path - 1*.

2.7.3 Directory structure of a target Datareplicator

Figure 2-6 shows a target Datareplicator's directory structure, and Table 2-16 lists the file contents.

Figure 2-6: Target Datareplicator's directory structure (Windows)



#

Environment variable that indicates the directory that stores the Datareplicator definitions.

Table 2-16: Contents of directories and files created by a target Datareplicator (Windows)

Creation time	Directory and filename ^{#1, #2}	Description
Created by the user	%HDSPATH%\hdsenv	Import system definition file
	<i>any-name</i>	Import environment definition files
	<i>any-name</i>	Import definition files
	<i>file-name-specified-in-file_dupenv-oper and-in-extraction-system-definition-file</i>	Duplexing definition file
Target Datareplicator's initial startup	<i>any-name</i>	Import information queue files
	<i>any-name</i>	Import status files
	%HDSPATH%\hdsinitstate	Import master status file
	%HDSPATH%\errfile1 %HDSPATH%\errfile2	Import error information files
	%HDSPATH%\reftrc.trc1 %HDSPATH%\reftrc.trc2	Import trace files
	<i>any-name</i>	Unimported information files
	<i>any-name</i> ^{#3}	Command log files
Created by the user	<i>any-name</i>	Update information definition file ^{#4}
Transferred by the user	<i>any-name</i>	SAM file ^{#5}
During execution of the update information input command	<i>update-information-input-command-execution-directory\unextfile_data-linkage-identifier</i>	Unextracted data storage file ^{#6}

#1: Each directory is created under a local drive.

#2: The HDSPATH environment variable specifies the directory used to create the target Datareplicator's definitions.

#3: Specify the absolute or relative path name of the command log file in the hds_command_log_file environment variable. If the relative path name is specified, the system assumes %HDSPATH%\relative-path-name as the absolute path name. If you omit the hds_command_log_file environment variable, the system assumes %HDSPATH%\hdscmdlog as the name of the command log file. The actual command log file name has 1 or 2 appended at the end of the specified name. Make

sure that the length of the actual command log file name does not exceed the *maximum length of the OS path - 1*.

#4: Of mainframe databases when a SAM file is used, you must create an update information definition file for a target Datareplicator that extracts PDM2 E2 data. There is no need to create this file for any other Datareplicator.

#5: The update information extraction SAM file for the mainframe is sent by the file transfer program. You must create a SAM file when the target Datareplicator is linked to a mainframe database that uses SAM files (PDM2 E2 or RDB1 E2). There is no need to create this file for any other Datareplicator.

#6: The unextracted data storage file is created/re-created each time an update information input command is executed. Its filename is `unextfile_` plus the data linkage identifier specified during execution of the update information input command.

2.8 Specifying environment variables (Windows)

This section explains how to specify the system environment variables for the source Datareplicator and for the target Datareplicator and the values to be specified.

To specify an environment variable:

1. In **Control Panel**, double-click the **System** icon.
2. In the System dialog box, choose the **Environment** tab, specify values under **Variable** and **Value**, and then click the **Set** button. Set only system environment variables.

If you specify environment variables after you have installed Datareplicator, you must restart Windows. To avoid having to do this, specify the environment variables before you install Datareplicator.

When you check environment variables, use **Control Panel** to check the system environment variables.

2.8.1 Environment variables for a source Datareplicator

The following table lists and describes the environment variable settings for a source Datareplicator.

Table 2-17: Information specified in the environment variables for a source Datareplicator

Environment variable	Description
PATH	Add the following paths: <ul style="list-style-type: none"> • <i>source-Datareplicator's-installation-directory</i>\bin^{#1} • %PDDIR%\bin^{#2} • %PDDIR%\client\utl^{#2}
TZ	Specify the time zone of the messages output from the source Datareplicator. This value must match the comparable valued specified in the HiRDB system definition.
PDDIR ^{#3, #4}	Specify the source HiRDB directory.
PDCONFPATH ^{#3, #4}	Specify the directory for storing the source HiRDB's system definition files.
PDNAMEPORT ^{#3}	Specify the source HiRDB's port number.
PDHOST ^{#3}	Specify the source HiRDB's host name.
PDUSER	Specify the user connected to the source HiRDB.

Environment variable	Description
HDEPATH	<p>Specify the source Datareplicator's operation directory. The source Datareplicator's definition files, status files, and error information files are created in this directory. The source Datareplicator's core file (complete processing dump file) is also output to this directory.</p> <p>In Windows, you can use parentheses in the operation directory name.</p> <p><i>Notes:</i></p> <ul style="list-style-type: none"> • If the source and target Datareplicators are located on the same machine, do not specify the same directory in the HDEPATH and HDSPATH environment variables. • If the source HiRDB is a parallel server consisting of multiple server machines, specify the same operation directory for the source Datareplicator at each server machine.
HDE_BIN_COL_MAXLEN	<p>Specify a value (kilobytes) that is smaller than the actual definition length of any BLOB-type column that is handled internally in Datareplicator. If a BLOB-type column has a definition length of 2 GB or greater but its actual data is small, this environment variable enables the column data to be linked without having to redefine the table.</p> <p><i>Notes:</i></p> <ul style="list-style-type: none"> • The definition length of each BLOB-type column must be 262,143 KB or less. • The value specified in this environment variable applies to all BLOB-type columns that are subject to extraction. • For details about the handling of BLOB-type columns whose definition length exceeds the value specified in this environment variable, see 6.4.3(4) <i>Notes about the HDE_BIN_COL_MAXLEN environment variable.</i>

#1: This environment variable is set as the name of the command library during installation; for details, see 2.6.4 *Information registered during installation.*

#2: If the source HiRDB has been set up using the default settings, this environment variable is set during installation. If the source HiRDB has been set up using the settings with identifiers, make sure that you specify this environment variable.

#3: This is a source HiRDB environment variable. For details about the HiRDB design, see the *HiRDB Version 9 Installation and Design Guide.*

#4: If the source HiRDB is a parallel server and this setting varies from one server to another, the extraction system definition can be used to define this information for each server machine. If definitions for a specific server are omitted from the extraction system definition, the corresponding server uses the values of the environment variables that were specified for the user environment where the source Datareplicator's commands are executed.

2.8.2 Environment variables for a target Datareplicator

The following table lists and describes the environment variable settings for a target Datareplicator.

Table 2-18: Information specified in the environment variables for a target Datareplicator

Environment variable	Description
PATH	Specify the name of the target Datareplicator's commands library. ^{#1}
TZ	Specify the time zone of the messages output from the target Datareplicator. This value must match the comparable value specified in the HiRDB system definition.
PDDIR ^{#2}	Specify the target HiRDB directory.
PDNAMEPORT ^{#2}	Specify the target HiRDB's port number.
PDHOST ^{#2}	Specify the target HiRDB's host name.
HDSPATH	Specify the target Datareplicator's operation directory. The target Datareplicator's definition files, status files, and error information files are created in this directory. The target Datareplicator's core file (complete processing dump file) is also output to this directory. In Windows, you can use parentheses in the operation directory name. <i>Note:</i> <ul style="list-style-type: none"> If the source and target Datareplicators are located on the same machine, do not specify the same directory in the HDEPATH and HDSPATH environment variables.
HDS_MST_STDCLOSE	Specify whether the standard input, standard output, and standard error output are to be closed by the target Datareplicator. When this environment variable is omitted or TRUE is specified: The standard input, standard output, and standard error output will be closed by the target Datareplicator. When FALSE is specified: The standard input, standard output, and standard error output will not be closed by the target Datareplicator. When you specify FALSE, an import information editing UOC routine can use the standard input, standard output, or standard error output. If an import information editing UOC routine uses the standard input, standard output, or standard error output when this environment variable is not specified, data might not be imported correctly. If you plan to have an import information editing UOC routine use the standard input, standard output, or standard error output, make sure that you specify FALSE in this environment variable.
HDS_RFI_ELANG ^{#3}	Specify the character encoding used in the source system. If this information has not been specified, the system automatically identifies the character encoding on the basis of the import status file. This automatic identification is not available when the definition information is displayed immediately after the target system has been initialized. If automatic identification is not available, ja_JP.SJIS is assumed.
HDS_RFI_PLANG ^{#3}	Specify the character encoding used in the target system. If this information has not been specified, ja_JP.SJIS is assumed.

Environment variable	Description
HDSCLTWAITTIME	<p>Specify whether the values of PDCWAITTIME (client's maximum wait time) and PDSWAITTIME (server's maximum wait time)^{#4} specified in the HiRDB client environment definition are to take effect on the target Datareplicator.</p> <p>This environment variable is optional; we recommend that you omit it if it is not needed.</p> <p>USER Uses the values specified in the HiRDB client environment definition.</p> <p>REPL Ignores the values specified in the HiRDB client environment definition and sets the following values:</p> <ul style="list-style-type: none"> • PDCWAITTIME Sets 0 (wait until there is a response). • PDSWAITTIME Sets the commit_wait_time operand value in the import system definition^{#5} + 600 (seconds). If the import transaction synchronization facility is used, the syncwait_limit_tim operand value in the import system definition + 600 (seconds) is set. If the specified setting exceeds 65535, 65535 is set. <p>When this environment variable is omitted: If an import information editing UOC routine is used, USER is assumed. If the import SQL process is used, REPL is assumed.</p> <p>When USER is specified: Set in PDCWAITTIME and PDSWAITTIME in the client environment definition values that are greater than the following operand values:</p> <ul style="list-style-type: none"> • PDCWAITTIME pd_lck_wait_timeout operand in HiRDB • PDSWAITTIME commit_wait_time and syncwait_limit_time operands in the import system definition <p>If the values set in PDCWAITTIME and PDSWAITTIME are too small, HiRDB's monitoring interval might be reached. In such a case, the target Datareplicator's SQL process detects an SQL error, resulting in abnormal termination.</p>

#1

The name of the commands library is specified during installation; for details, see *2.6.4 Information registered during installation*.

#2

This is a target HiRDB environment variable. For details about the HiRDB design, see the *HiRDB Version 9 Installation and Design Guide*.

#3

The following table shows the values of HDS_RFI_ELANG and HDS_RFI_PLANG:

2. Environment Setup

Environment variable	Character encoding			
	JIS8/ Shift JIS	EUC	Unicode (UTF-8)	EBCDIC/KEIS EBCDIK/KEIS
HDS_RFI_ELANG	ja_JP.SJIS	ja_JP.UJIS	ja_JP.UTF8	EBCDIK
HDS_RFI_PLANG	ja_JP.SJIS	ja_JP.UJIS	ja_JP.UTF8	--

Legend:

--: Not applicable

#4

This does not apply to `PDSWATCHTIME` in the HiRDB client environment definition. The value specified in the HiRDB client environment definition always takes effect.

#5

If the `commit_wait_time` operand is specified in the import environment definition, the value specified in the import environment definition takes effect.

2.9 Setting up the communications environment (Windows)

Before you can execute Datareplicator, you must set up the communications environment. This section describes the settings for the communications environment.

2.9.1 Registering the service name

Specify the service name and port number in the `services` file (`Windows-system-directory\drivers\etc\services`) in the following format:
`service-name port-number /tcp`

(1) Specifications for a source Datareplicator

The following table shows the information that is specified in the `services` file for a source Datareplicator.

Table 2-19: Information specified in the `services` file for a source Datareplicator

Service name	Port number ^{#1}	Server machine subject to specification	
		Source HiRDB is a single server	Source HiRDB is a parallel server
Specify a service name to be used for communication with the target system. Specify this name also in the <code>hdeservice</code> operand in the transmission environment definition.	Specify a number available in the system. Specify the same number as in the target system.	Server machine at which the single server is defined	All server machines at which a back-end server is defined ^{#2}
Specify service name <code>hdenmserv</code> , which is to be used for communication with the source master process and the source node master process. You cannot change this name. Specify this name after completing installation or before terminating or restarting Windows.	Specify a number available in the system. Specify the same port number at the servers where the system manager and back-end servers are located.	Server machine at which the single server is defined	<ul style="list-style-type: none"> • Server machine at which the system manager is defined • All server machines at which a back-end server is defined²

#1: You cannot specify a port number that has already been registered in the `services` file or that is being used by other software.

#2: This includes back-end servers that do not contain any database subject to data extraction.

(2) Specifications for a target Datareplicator

The following table shows the information that is specified in the `services` file for a target Datareplicator.

Table 2-20: Information specified in the `services` file for a target Datareplicator

Service name	Port number [#]
Specify a service name to be used for communication with the source system. Specify this name also in the <code>hdsservice</code> operand in the import system definition.	Specify a number available in the system. Specify the same number as in the source system.

[#]: You cannot specify a port number that has already been registered in the `services` file or that is being used by other software.

2.9.2 Registering the host name

Register the host name to be used for communication with the target system in the `hosts` file (`Windows-system-directory\drivers\etc\hosts`) in the format shown as follows. Register this host name in the server machine where the single server is defined or in all server machines where a back-end server is defined.

IP-address host-name

Specify any address for *IP-address* and any name for *host-name*. You cannot specify an IP address or host name that has already been registered in the `hosts` file.

2.9.3 Using Windows Terminal Service

You can use Windows Terminal Service to run Datareplicator. Doing so enables you to run Datareplicator on remote machines and on machines without a console. For details about Windows Terminal Service, see the OS documentation.

Windows Terminal Service supports console sessions and virtual sessions.

- Console session

A console session switches the input and output devices (such as display, keyboard, and mouse) of a server to the devices of a client on the terminal server. Please note the following in particular:

- Start the command for establishing remote desktop connection to the server with the `/console` option specified.
- Only one session can be created in the client and the server (the previous window is disabled (logged off)).
- Console sessions are not supported in Windows 2000, Windows Vista, or Windows Server 2008.

- Virtual session

A virtual session runs a server in the background (creates the windows for running a client in the background). Please note the following in particular:

- Start the command for establishing remote desktop connection to the server without any options specified.
- Multiple virtual sessions can be created.

(1) Datareplicator versions and availability of Windows Terminal Service

Windows Terminal Service might not be available depending on the Datareplicator version. The following table shows the Datareplicator versions and whether Windows Terminal Service is supported.

Table 2-21: Datareplicator versions and whether Windows Terminal Service is supported

Datareplicator version	Console session	Virtual session
08-00 or earlier	WS2003	N
08-01 or later	Y	Y

Legend:

Y: Supported.

WS2003: Supported only in Windows Server 2003.

N: Not supported.

(2) Notes about security

When Windows Terminal Service is used, the client's window might be logged in, even if the server's console is not logged in. For this reason, the following server and client settings and handling are required:

- Use a screen saver password for protection.
- Log off when operations from the client have been completed.

2.10 Operating environment in Windows Vista and Windows Server 2008

The Datareplicator operating environment in Windows Vista and Windows Server 2008 might differ from other editions of Windows. This section explains the Datareplicator operating environment in Windows Vista and Windows Server 2008.

2.10.1 Executing commands

Windows Vista and Windows Server 2008 support a security feature called UAC (User Account Control). When UAC is used, ordinary tasks are performed with the minimum privilege. A higher privilege is granted only when a task requiring administrator privilege, such as the system setup, is performed. For details about UAC, see the OS documentation.

When a Datareplicator command is executed while UAC is used, the command processing depends on whether you run the command prompt as an administrator.

If you run the command prompt as an administrator:

When the command prompt is started, a confirmation dialog box is displayed asking whether the operation is to be permitted. If the response entered in the confirmation dialog box is that the operation is to be permitted, the command prompt is started and commands can be executed at the command prompt as usual.

If you do not run the command prompt as an administrator:

When the command prompt is started, no confirmation dialog box is displayed. Instead, a confirmation dialog box asking whether program access is to be permitted is displayed each time a command is executed at the command prompt. If program access is to be permitted pursuant to the response entered in the confirmation dialog box, the command can then be executed at the command prompt.[#]

#

Some commands will not execute successfully because the command prompt for the processing result might disappear. For this reason, we recommend that you execute commands by running the command prompt as an administrator.

2.10.2 Renamed commands

Some commands used in Windows Vista and Windows Server 2008 have names that differ from those used in the other OSs. The following table lists the commands whose names differ in Windows Vista and Windows Server 2008.

Table 2-22: Commands whose names differ in Windows Vista and Windows Server 2008

OS other than Windows Vista or Windows Server 2008	Windows Vista and Windows Server 2008
setup_tool1	set_tool1
setup_tool2	set_tool2
unsetup_tool	unset_tool

2.10.3 Support of JIS standard level 3 and level 4 character sets

To link data containing the JIS standard level 3 and level 4 character sets supported in Windows Vista and Windows Server 2008, the conditions listed below must all be satisfied.

Table 2-23: Conditions for linking data containing JIS standard level 3 and level 4 character sets (Windows only)

Environment	Condition
OS	Windows Vista or Windows Server 2008
Database	HiRDB version 08-02 or later
Datareplicator	Datareplicator version 08-01 or later
dblocale operand value in the extraction system definition	utf-8
dblocale operand value in the import system definition	

Note:

If any of these conditions is not satisfied, characters might be converted as undefined characters or as spaces or might not be displayed correctly.

2.11 Upgrading Datareplicator

This section explains how to upgrade Datareplicator. You must initialize Datareplicator in order to upgrade it.

For details about how to downgrade Datareplicator, see Appendix *D. Downgrading Datareplicator*.

2.11.1 Notes about upgrading

If you are using a Datareplicator at a parallel server, you must use the same Datareplicator version at the system manager and at all back-end servers.

2.11.2 How to upgrade

This subsection explains how to upgrade Datareplicator. This procedure assumes that you have already upgraded your HiRDB.

To upgrade Datareplicator:

1. Terminate HiRDB and Datareplicator normally. If you are using a HiRDB/Parallel Server, terminate all servers normally.
2. Install the new version of Datareplicator. To inherit the contents of various Datareplicator files (such as definition files and status files), check the definitions.

2.11.3 Notes on upgrading

Note the following once you have upgraded your Datareplicator:

- Make sure that you execute an initial start the first time you start Datareplicator after upgrading.
- In UNIX, even if you have upgraded your Datareplicator to the 64-bit edition, you will still preprocess, compile, and link UOC routines in the 32-bit environment; Datareplicator does not support executable files using a 64-bit library. If you attempt to use such a UOC routine, the results might not be correct.

2.12 Improving the reliability of syslogfile and character encoding conversion (Linux only)

If you will be performing data linkage in the Linux edition, you can apply the extended SYSLOG facility.[#] Installing and then applying the extended SYSLOG facility will improve the reliability of `syslogfile` and will enable you to perform character encoding conversion on `syslogfile`.

#

The extended SYSLOG facility is a program provided by the Linux support service (SD-LS100-FR1N1 or SD-LS200-FR1N1).

2.12.1 Improving the reliability of syslogfile

If a memory shortage occurs as a result of output of a large volume of messages to `syslogfile`, some messages might be lost. If output of messages to `syslogfile` fails, message output is retried so that loss of messages will not occur.

Prerequisites

The following table shows the versions of Linux and the extended SYSLOG facility that support this facility.

Table 2-24: Correspondence between Linux versions and the extended SYSLOG facility

Linux version	Extended SYSLOG facility version
RedHat Enterprise Linux 4 Update 3 (IPF)	01-00 or later
RedHat Enterprise Linux 4.5 (IPF)	01-01 or later
RedHat Enterprise Linux 4 Update 3 (x86)	01-02 or later
RedHat Enterprise Linux 4.5 (x86)	
RedHat Enterprise Linux 4.5 (EM64T)	
RedHat Enterprise Linux 5.1 (IPF)	02-00 or later
RedHat Enterprise Linux 5.1 (x86)	
RedHat Enterprise Linux 5.1 (EM64T)	

2.12.2 Character encoding conversion on syslogfile

You can change the character encoding of messages that are output to `syslogfile` from SJIS to UTF-8.

Note that character encoding conversion cannot be applied to messages related to operations management that are output to `syslogfile` from processes using JP1/Cm2 and the processes of Datareplicator's operation commands.

Character encoding conversion sets to UTF-8 the character encoding for messages that are output to `syslogfile`, which results in the following benefits:

- It is easier to monitor and manage messages.
- When `syslogfile` is viewed, there will be no garbled characters in the messages.

Prerequisites

To perform character encoding conversion on `syslogfile`, both the following conditions must be satisfied:

- The combination of the Linux, extended SYSLOG facility, and Hitachi encoding conversion component versions must match the applicable combination shown in the following table.

Table 2-25: Versions of Linux, extended SYSLOG facility, and Hitachi encoding conversion component

Linux version	Version of extended SYSLOG facility	Version of Hitachi encoding conversion component
RedHat Enterprise Linux 5.1 (IPF)	Extended SYSLOG facility 02-00 or later	Hitachi encoding conversion component - Runtime (64) 02-03 or later
RedHat Enterprise Linux 5.1 (x86)	Extended SYSLOG facility 02-01 or later	Hitachi encoding conversion component - Runtime 02-03 or later
RedHat Enterprise Linux 5.1 (EM64T) 32-bit addressing mode	Extended SYSLOG facility 02-01 or later	Hitachi encoding conversion component - Runtime 02-05 or later
RedHat Enterprise Linux 5.1 (EM64T) 64-bit addressing mode		Hitachi encoding conversion component - Runtime (64) 02-05 or later

- `sjis` is specified in the `dblocale` operand in the extraction system definition and the import system definition, and `sjis-japanese` or `english` is specified in the `msglocale` operand.

2.12.3 Notes

- Before you start Datareplicator, install the extended SYSLOG facility (if you will be performing character encoding conversion on `syslogfile`, also install the Hitachi encoding conversion component). If the extended SYSLOG facility is

installed while Datareplicator is running, this facility will not be applied.

- If the extended SYSLOG facility has been installed, the extraction master process and the extraction node master process output the KFRB00562-I message to `syslogfile` when the import master process starts.

If you will be using an import information editing UOC routine, see *8.1.6(9) Notes about the extended syslog facility*.

Chapter

3. Data Linkage Facilities

This chapter describes the data linkage patterns supported by Datareplicator, the facilities for data linkage extraction and import processing, and the functions provided by Datareplicator.

- 3.1 Linkage patterns
- 3.2 Source Datareplicator's extraction processing
- 3.3 Target Datareplicator's import processing
- 3.4 Using JP1/Cm2 for operations management
- 3.5 Datareplicator file system areas
- 3.6 Delay monitoring facility
- 3.7 Import transaction synchronization facility
- 3.8 Event facility
- 3.9 Duplexing files

3.1 Linkage patterns

This section discusses the patterns that can be linked with Datareplicator. These linkage patterns include data linkage for an abstract data type. For details about designing linkage patterns, see 4.2 *Designing a linkage pattern*. The explanations in this chapter are based on examples of importing data between relational databases.

3.1.1 Linking data to a table with the same format

This section explains by way of examples data linkage to a table with the same format.

(1) Importing into a table with the same format

The following figure shows an example of importing a source table into another table with exactly the same format (same table format, table name, and column names).

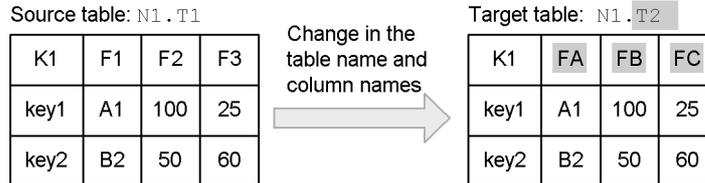
Figure 3-1: Example of importing into a table with the same format



(2) Importing into a table and changing only the table name or column names

The following figure shows an example of importing data and changing the source table's table name or column names.

Figure 3-2: Example of importing data and changing the table name or column names



■ : Not the same in the source and target tables

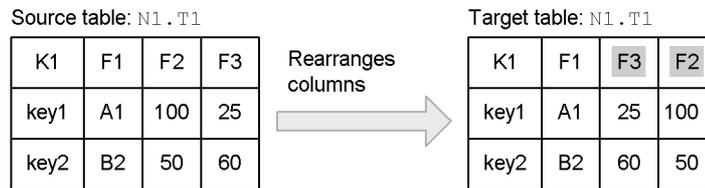
3.1.2 Linking data to a table with a different format

This section explains by way of examples data linkage to a table with a different format.

(1) Rearranging the columns

The following figure shows an example of importing data and rearranging the source table's columns. To rearrange columns, you define the linkage pattern either in the source system or the target system.

Figure 3-3: Example of importing data and rearranging the columns

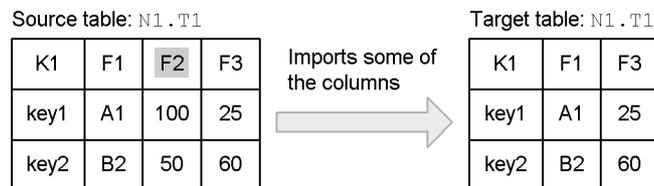


■ : Not the same in the source and target tables

(2) Importing selected columns

The following figure shows an example of importing selected columns from the source table. To import selected columns, you define the linkage pattern either in the source system or the target system.

Figure 3-4: Example of importing selected columns

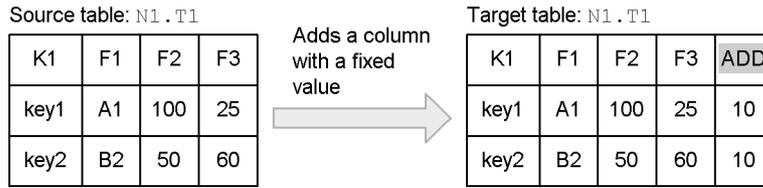


■ : Not the same in the source and target tables

(3) Adding a column with a fixed value

The following figure shows an example of adding a column with a fixed value during data import processing. To add a fixed-value column, you define the linkage pattern in the target system.

Figure 3-5: Example of adding a column with a fixed value



■ : Not the same in the source and target tables

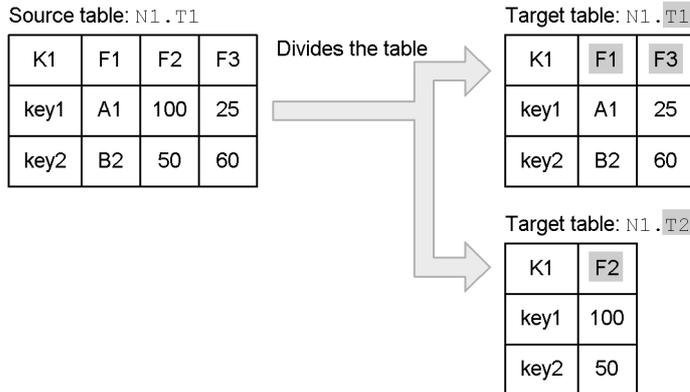
3.1.3 Linking data from one table to n tables

This section explains by way of an example data linkage from a single table to multiple tables.

(1) Dividing into multiple tables

The figure below shows an example of importing data and dividing the source table into multiple tables. To divide a table into multiple tables, you define the linkage pattern either in the source system or the target system. You can also import data segments into multiple destinations on the basis of a definition in the source system.

Figure 3-6: Example of dividing a table and importing into multiple tables



■ : Not the same in the source and target tables

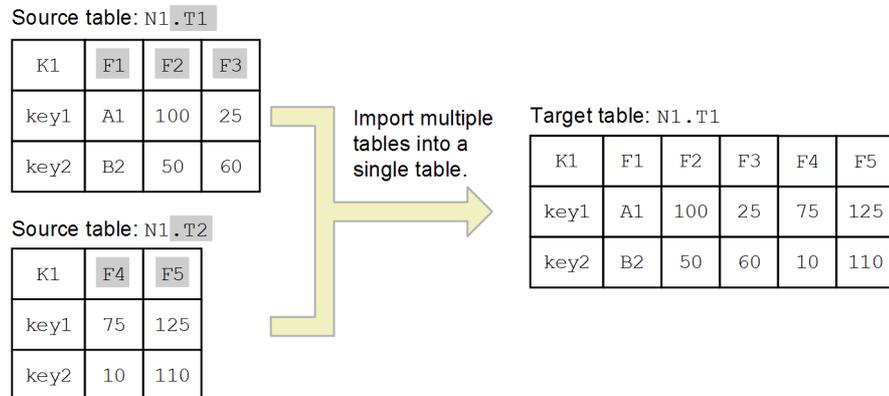
3.1.4 Linking data from n tables to one table

This section explains by way of an example data linkage from multiple tables to a single table.

(1) Importing multiple tables into a single table

The figure below shows an example of importing multiple tables into a single table. To import multiple tables into a single table, you define the linkage pattern in the target system.

Figure 3-7: Example of importing multiple tables into a single table



Legend:

■ : Not the same in the source and target tables

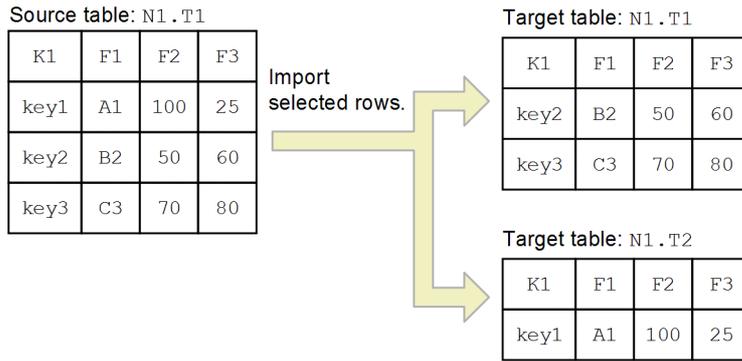
3.1.5 Linking data by selecting the rows to be sent

This section explains by way of an example data linkage by selecting the rows to be sent.

(1) Selecting the rows to be sent

This linkage pattern imports the update information in the source system that satisfies specified conditions. You can specify the conditions for each target system involved using the `send` statement in the extraction definition. The following figure shows an example of importing data by selecting the rows to be sent.

Figure 3-8: Example of importing data by selecting the rows to be sent



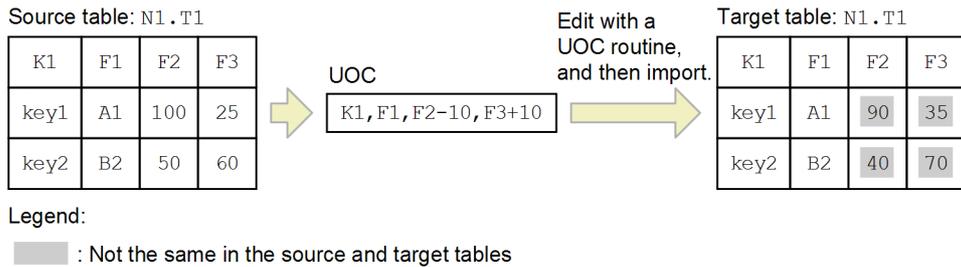
3.1.6 Linking data by using a user own coding routine

This section explains by way of examples data linkage by editing data with a UOC routine before importing it. For details about UOC routines, see Chapter 8. *User Own Coding Routines*.

(1) Editing update information with a UOC routine

The following figure shows an example of using an import information editing UOC routine to edit update information, and then importing the update information.

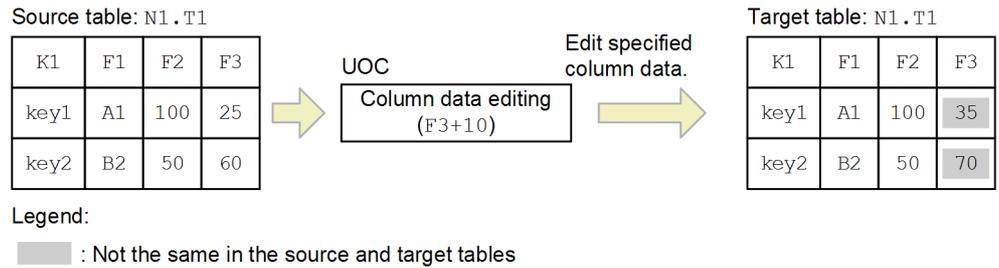
Figure 3-9: Example of editing update information with a UOC routine before importing it



(2) Editing column data with a UOC routine

The following figure shows an example of using a column data editing UOC routine to edit update information for columns, and then importing the update information.

Figure 3-10: Example of editing columns with a UOC routine before importing



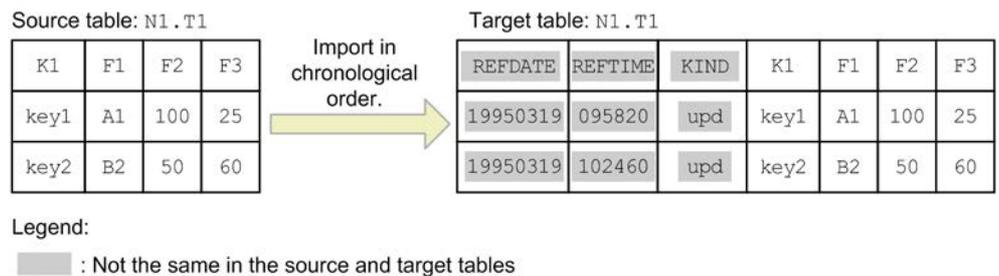
3.1.7 Obtaining a record of updates in chronological order

This section explains by way of an example data linkage by obtaining a record of updates in chronological order.

(1) Importing the import dates, times, and other information in chronological order in addition to the update information

The figure below shows an example of importing the import dates, times, and other information in chronological order in addition to the update information. To obtain time-ordered information, you define the linkage pattern in the target system.

Figure 3-11: Example of obtaining the import dates, times, and other information in chronological order in addition to the update information



(1) Extraction

The source Datareplicator extracts the update information from the source HiRDB's system log file and stores it in an *extraction information queue file*. If the source HiRDB is a parallel server, the source Datareplicator executes the extraction processing for each back-end server subject to extraction because there is a HiRDB system log file for each back-end server. In this case, Datareplicator stores the extraction status in the data linkage file and the extract-time status file.

If an error occurs during extraction processing, Datareplicator outputs error information to an error information file.

(2) Transmission

The source Datareplicator reads the update information in the extraction information queue file and sends it to the specified target system. If the source HiRDB is a parallel server, Datareplicator executes transmission for each applicable back-end server. In such a case, Datareplicator stores the transmission status in an extract-time status file.

If an error occurs during transmission, Datareplicator outputs the error to an error information file.

3.2.2 Files and processes used during extraction processing

This section explains the files and processes that are used during extraction processing. For details about the files and processes used when

JP1/Cm2 is used for operations management, see *3.4.5 Files and processes used for operations management*.

(1) Files used during extraction processing

The files discussed as follows are used during extraction processing.

(a) HiRDB system log file

This file stores database update information at the HiRDB. The source Datareplicator extracts update information from this file.

(b) Extraction information queue files

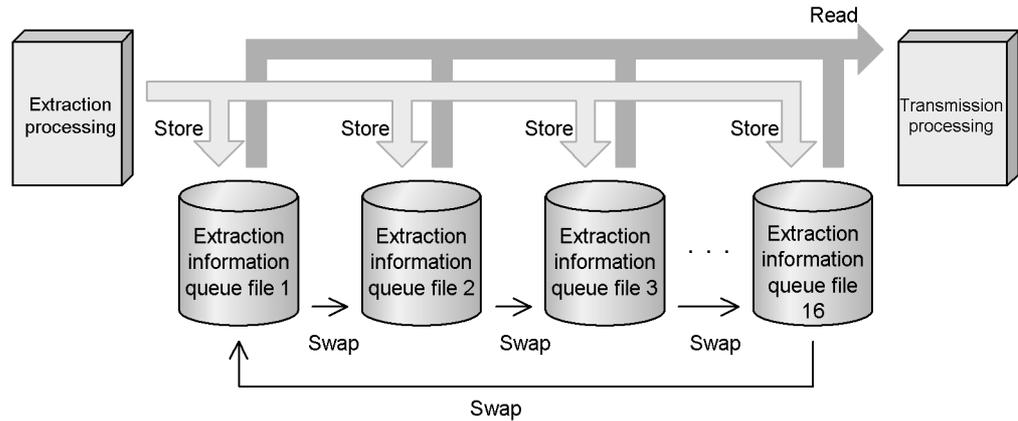
Datareplicator uses these files to store the update information extracted from the HiRDB system log file during extraction processing.

The source Datareplicator stores the update information extracted during extraction processing sequentially into one of the extraction information queue files. When this extraction information queue file becomes full, Datareplicator uses another extraction information queue file. This is called swapping, and it enables the source Datareplicator to store a large amount of update information. Swapping takes place in the order of the `qufile001` to `qufile016` operands specified in the extraction environment definition.

When all extraction information queue files become full, Datareplicator re-uses the first file. However, if transmission of the update information has not been completed for the file that is to be used next, swapping cannot take place. In such a case, Datareplicator outputs a message indicating that the queue file is full and stops extracting update information from the system log file until transmission from the file has been completed.

The following figure shows the procedure for storing data in the extraction information queue files.

Figure 3-13: Procedure for storing data in the extraction information queue files



(c) Data linkage file

Datareplicator uses the data linkage file to store and read HiRDB communication messages that are required for extraction processing, such as the storage status of update information in the system log at HiRDB and the status of reading update information from the system log file at the source Datareplicator.

(d) Extract-time status files

These files store the extraction/transmission status required for recovery in the event of an error. The extract-time status files include the extraction master status file and the extraction server status file.

(e) Extract-time error information files

If extraction or transmission processing results in an error, Datareplicator outputs error details to an error information file. The extract-time error information files include the extraction master error information files and the extraction node master error information files.

You can also output to the syslog file the information that is output to the error information files. You use the `syslogout` operand in the extraction system definition

to specify whether the information is to be output also to the syslog file.

(f) Extract-time activity trace file

Datareplicator uses the activity trace files to collect Datareplicator's activity status. These files contain information about Datareplicator's operation and performance. The activity trace files include the extraction master trace files and the extraction node master trace files.

To use the activity trace files, specify the `int_trc_lvl` and `int_trc_filesz` operands in the extraction system definition. You can edit the obtained activity trace files with the `hdstrcredit` command. For details about the `hdstrcredit` command, see its command syntax in Chapter 7. *Command Syntax*.

(g) Extraction system definition file

The extraction system definition file defines the overall operating environment for the source Datareplicator, such as the source Datareplicator's identifier and the target identifier.

(h) Extraction environment definition file

This file defines the operating environment for extraction processing, such as the names and sizes of the extraction information queue files.

(i) Transmission environment definition files

These files define operating environments for transmission processing, such as the service names and host names for communications.

(j) Extraction definition file

This file defines detailed information about extraction and transmission processing, such as the correspondence between source table/columns and update information and the destination of update information.

(k) Extraction definition preprocessing file

This file is obtained by using the `hdeprep` command to convert the extraction definition file to the internal format. You must execute this conversion to the extraction definition preprocessing file before you start the source Datareplicator.

Checking the validity of the extraction definition preprocessing file

When the source Datareplicator is started, the validity of the extraction definition preprocessing file is checked automatically. Startup processing of the source Datareplicator is cancelled in the following cases:

- The creation date of the extraction definition preprocessing file is earlier than the creation date of the extraction master status file (the `KFRB00713-E` message is output).
- The table definition for the table to be extracted had been changed after the

hdeprep command was executed (the KFRB00866-E message is output).

Note that if Datareplicator cannot connect to HiRDB for a reason such as omission of the PDUSER environment variable or a password, the KFRB00868-W message is output, in which case the source Datareplicator is started without checking the validity of the extraction definition preprocessing file.

(1) Command log files

These files store a record of the dates and times Datareplicator's commands are executed.

(2) Organization of processes during extraction processing

Figure 3-14 shows the organization of processes during extraction processing when the source HiRDB is a single server, and Figure 3-15 shows the organization of processes when the source HiRDB is a parallel server.

Figure 3-14: Organization of processes during extraction processing: Source HiRDB is a single server

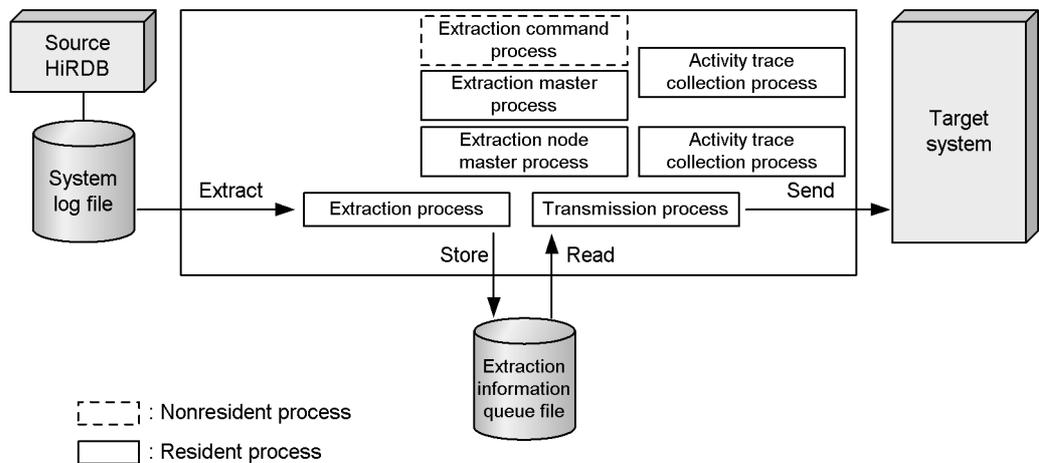
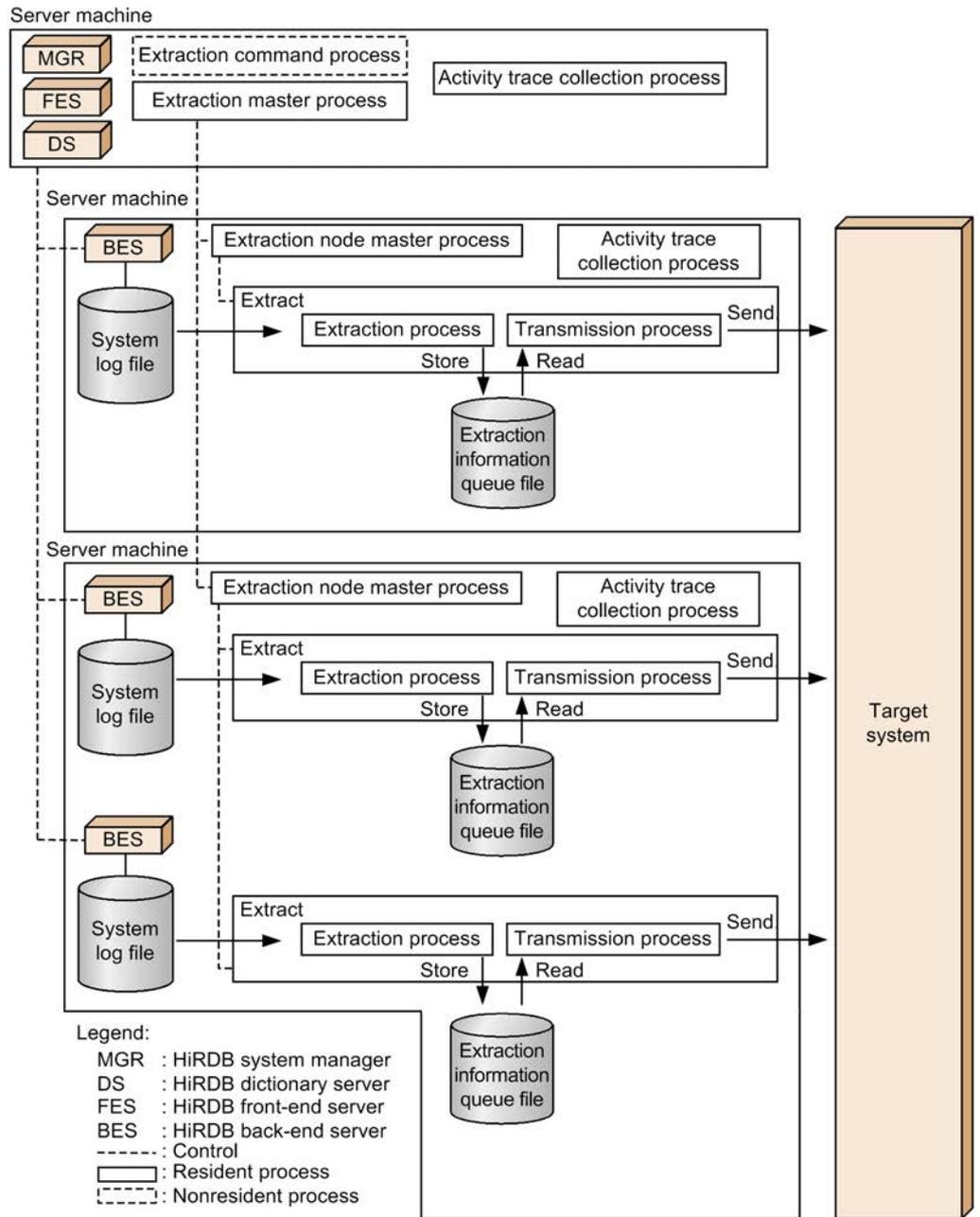


Figure 3-15: Organization of processes during extraction processing: Source HiRDB is a parallel server



(a) Extraction command process

The extraction command process processes the source Datareplicator's command and issues an instruction to the extraction master process. If the source HiRDB is a parallel server, there is one extraction command process under the system manager.

(b) Extraction master process

The extraction master process controls the extraction node master process. If the source HiRDB is a parallel server, there is one extraction master process under the system manager.

(c) Extraction node master process

The extraction node master process manages the extraction process and the transmission process. If the source HiRDB is a parallel server, there is one extraction node master process at each server machine that contains a back-end server. The source Datareplicator calls such a server machine a node.

(d) Extraction process

The extraction process extracts update information from the system log file and stores it in the extraction information queue file. If the source HiRDB is a parallel server, one extraction process exists at each back-end server that is subject to extraction.

(e) Transmission process

The transmission process reads update information from the extraction information queue file and sends it to the target system. There are as many transmission processes as there are destinations. If the source HiRDB is a parallel server, there are as many transmission processes as there are destinations at each back-end server that is subject to extraction. However, if you specify `sendmst` in the `sendcontrol` extraction system definition operand, up to the number of transmission process can exist as is specified in the `sendprocnum` extraction system definition operand.

Hereafter, the method used when `sendmst` is specified in the `sendcontrol` extraction system definition operand is referred to as the `sendmst` method, while the method used when `nodemst` is specified is referred to as the `nodemst` method.

(f) Transmission master process

The transmission master process starts, stops, and schedules the transmission processes to control/suppress the number of transmission processes to be started when there are many destinations. The transmission master process is generated when a definition is made to control the number of transmission processes. For details about how to control the number of transmission processes, see *3.2.5 Controlling the number of transmission processes*.

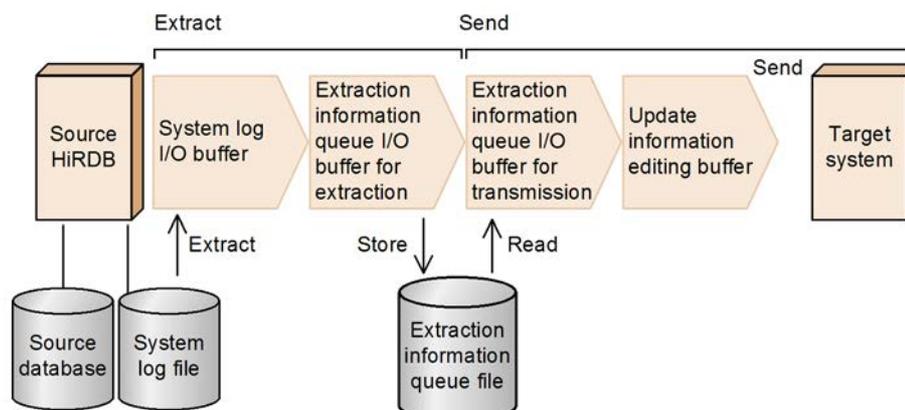
(g) Activity trace collection process

The activity trace collection process collects activity trace information.

3.2.3 Units of extraction processing

The following figure shows the units of extraction processing.

Figure 3-16: Units of extraction processing



(1) Extraction

The source Datareplicator reads the update information stored in the system log file and stores it in an extraction information queue file according to the extraction definition. To do this, Datareplicator uses the system log I/O buffer and extraction information queue I/O buffer for extraction. If the source HiRDB is a parallel server, Datareplicator executes extraction processing for each back-end server. For details about each buffer used for extraction, see 4.6.3 *Designing the extraction procedure*.

(2) Transmission

The source Datareplicator reads the update information stored in the extraction information queue file for each destination, edits this information if the transaction has been completed, and then sends it to the target system. To do this, Datareplicator uses the extraction information queue I/O buffer for transmission and the update information editing buffer. For details about each buffer used for transmission, see 4.6.4 *Designing the transmission procedure*.

3.2.4 Collecting information about the source Datareplicator

Commands and system definitions enable you to obtain the information about the source Datareplicator that is explained as follows.

(1) Status information

By executing the status information collection command (`hdestate` command), you can output *status information* to the standard output. This information can be used to check the source Datareplicator's status. You can collect status information only while

the source Datareplicator is active.

The status information includes the source Datareplicator's identifier, node information, and extraction information queue file utilization status. For details about how to collect status information, see the *hdestate* command (collect source Datareplicator's status information) in Chapter 7. *Command Syntax*.

(2) Activity trace

You can collect an *activity trace* to obtain Datareplicator's activity status and to determine its performance. You can use an activity trace for the following purposes:

- If extraction/transmission processing seems slow, use the activity trace for guidance in changing system settings to maximize HiRDB performance (for example, for determining whether tuning should involve changing the extraction buffer size, changing the send buffer or transmission interval, or increasing the communication line capacity).
- If a send data UOC routine's performance is not satisfactory, use the activity trace to determine whether the problem is with Datareplicator or with the UOC routine.

To obtain an activity trace with the source Datareplicator, you specify the `int_trc_lvl` and `int_trc_filesz` operands in the extraction system definition. To obtain an activity trace for each back-end server, you specify the `int_trc_getv` operand in the extraction environment definition. To obtain an activity trace for each target system identifier, you specify the `int_trc_getv` operand in the transmission environment definition. Use the `hdstrcredit` command to output an obtained activity trace.

For details about activity trace definitions, see 5.2 *Extraction system definition*, 5.3 *Extraction environment definition*, and 5.4 *Transmission environment definition*; for details about the `hdstrcredit` command, see the *hdstrcredit* command in Chapter 7. *Command Syntax*.

3.2.5 Controlling the number of transmission processes

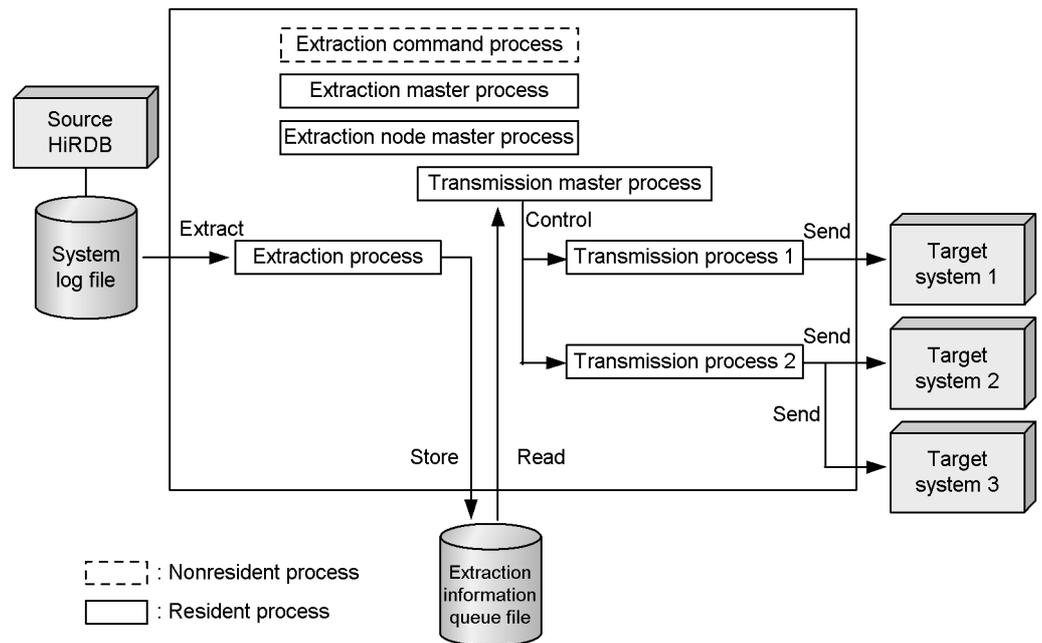
Datareplicator starts a transmission process for each destination. If there are many destinations, there will be many transmission processes and a memory shortage might occur in the source system, or queue file I/O operations might slow down. To prevent this, you can control the transmission processes to be started so that all transmission processing is executed within a specified number of transmission processes.

To control the number of transmission processes, specify `sendmst` in the `sendcontrol` operand in the extraction system definition. When you make this specification, a transmission master process is generated between the node master process and the transmission processes. This transmission master process controls the startup, termination, and transmission schedule of the transmission processes.

The following figure provides an overview of transmission process control. This

example uses two transmission processes to achieve transmission to three destinations.

Figure 3-17: Overview of transmission process control



(1) Function of the transmission master process

The transmission master process reads update information from the extraction information queue file before the transmission interval is reached and creates valid transaction management information for each destination. If the transmission master process successfully creates valid transaction management information within the transmission interval, it issues transmission requests to the transmission processes. The transmission master process then schedules the active transmission processes within the constraints of the maximum number of active processes.

(2) Function of a transmission process when the number of transmission processes is controlled

When the number of transmission processes is controlled, the function of a transmission process varies as follows:

- The transmission process reads update information from the extraction information queue file when it receives transmission management information from the transmission master process.
- After sending the update information, the transmission process reports

completion of transmission to the transmission master process and stays active while it waits for another transmission request from the transmission master process.

3.2.6 Processing update information with a user own coding routine

To aid data linkage applications, you can create the following user own coding (UOC) routine for the source Datareplicator:

- Send data UOC routine

This UOC routine enables you to check update information to decide whether to send it.

For details about UOC routines, see Chapter 8. *User Own Coding Routines*.

3.2.7 Suppressing message output

While it is operating, Datareplicator outputs to the syslog file (or event log for Windows), and to error information files messages that report the activity status. These messages are important to determine the activity status; however, many messages might be output depending on the operating environment and they might use up a large amount of resources.

To minimize this problem, you can suppress output of some messages that you determine to be unnecessary. The following table shows the operands used to suppress message output:

Operand	Output destination subject to suppression
syslogout	syslog file (or event log)
syslog_message_suppress	
node_syslogout	
info_message_out [#]	<ul style="list-style-type: none"> • syslog file (or event log) • Error information files

Note

For details about each operand, see 5.2 *Extraction system definition*.

[#]: If you have specified `suppress` in the `info_message_out` operand to suppress output of specific messages, you can use the `except_suppress` operand to exclude any of the specified messages so that those messages will be output to the console and error information files.

The following table shows the results of message output when multiple operands are specified to suppress message output to the syslog file (or event log):

Operand			Message output destination	
info_message_output	syslogout or node_syslogout	syslog_message_suppress	syslog file (or event log)	Error information files
suppress	Not specifiable	Not specifiable	N	N
nosuppress	true	Specified	N	Y
		Omitted	Y	Y
	false	Not specifiable	N	Y

Y: Output.

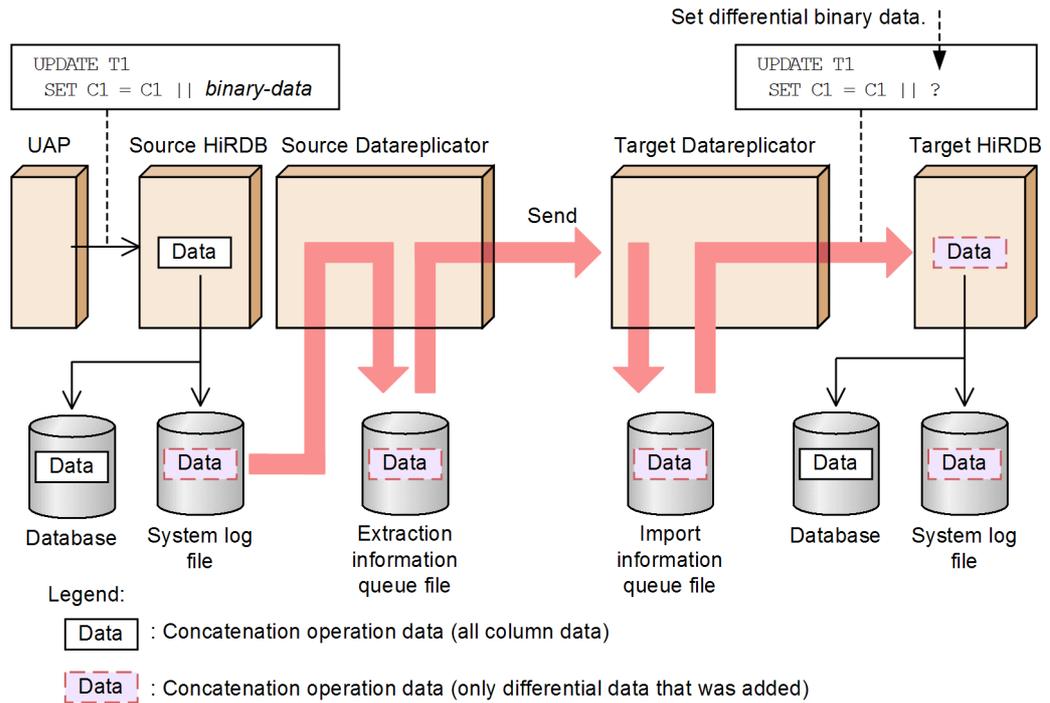
N: Not output.

3.2.8 Linking update data for concatenation operations

You can also use Datareplicator to link update data for concatenation operations. Updating of a concatenation operation results in an update log that consists only of the data part added by the concatenation operation. This means that only the differential data is extracted and imported, unlike the data linkage of conventional BLOB and BINARY columns that involves extraction and import of the entire columns.

The following figure shows linking of update data for concatenation operations.

Figure 3-18: Linking update data for concatenation operations



Note:

In Databrowser Version 6, concatenation operations were referred to as *acquisition of BLOB column partitions*.

3.2.9 Data linkage for RDAREAs using the inner replica facility

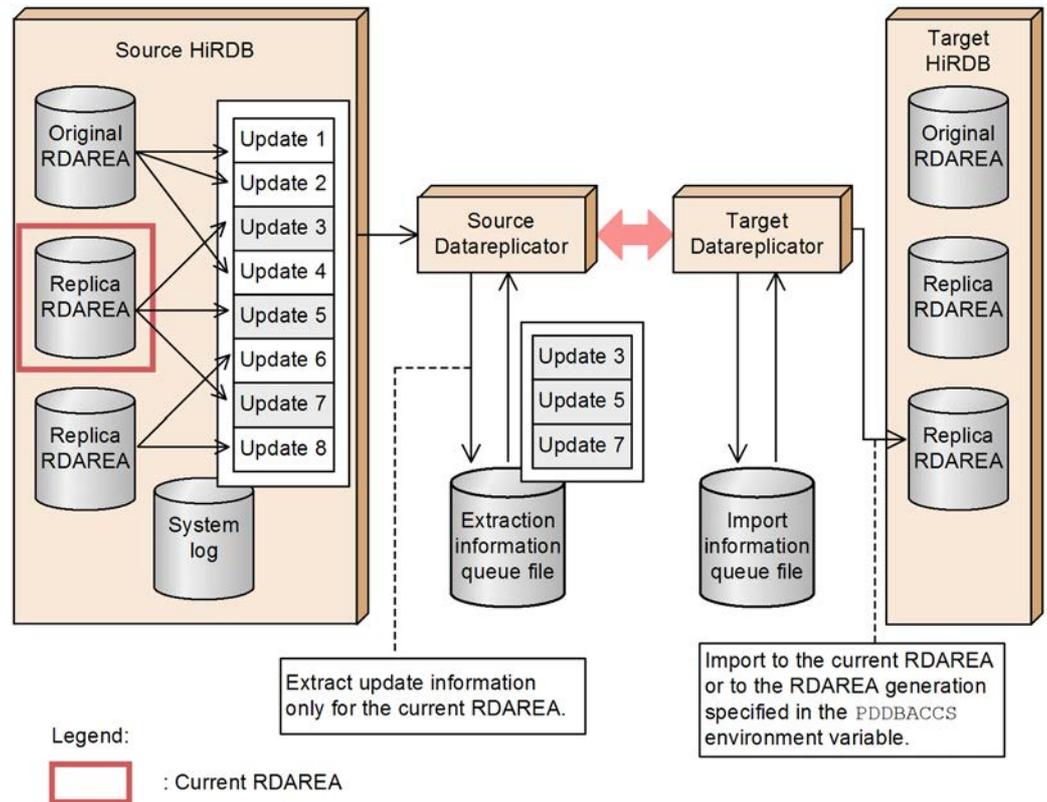
With Databrowser, you can also use HiRDB's *inner replica facility* to link data in RDAREAs.

When the inner replica facility is used, the update log is always output to the system log file whether the updated RDAREA is the original or a replica. Databrowser extracts from this update log the log information corresponding to the source table ID.

If the target database uses the inner replica facility, you can import the data into the current RDAREA that is subject to data linkage (original or replica RDAREA) or into a specific generation of the database. You cannot specify a different target for each table. To import data into a specific database generation, specify the database with HiRDB's `Pddbaccs` environment variable.

The following figure shows data linkage for RDAREAs using the inner replica facility.

Figure 3-19: Data linkage for RDAREAs using the inner replica facility



(1) Data linkage definitions for inner replica

To specify the method for extracting update log information, use the `extract_level` operand described in 5.3 *Extraction environment definition*. The relationship between the `extract_level` operand value and the update log information to be extracted is as follows:

- `current_gen`: When the source HiRDB uses the inner replica facility, extract only the updating of the current RDAREA.
- `all_gen`: When the source HiRDB uses the inner replica facility, also extract the updating of all replica RDAREAs (including the original RDAREA).

Use the following guidelines to specify the `extract_level` operand based on the handling of inner replica:

3. Data Linkage Facilities

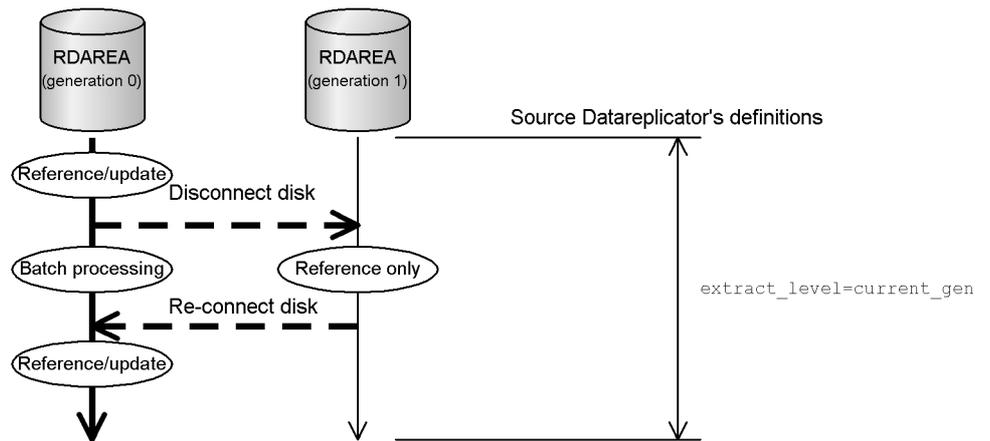
Handling of inner replica	extract_level operand value
The current RDAREA is always the master (source subject to extraction).	Specify <code>extract_level=current_gen</code> .
An RDAREA other than the current RDAREA can be the master (source subject to extraction).	Specify <code>extract_level=all_gen</code> when the disk is disconnected [#] .

#

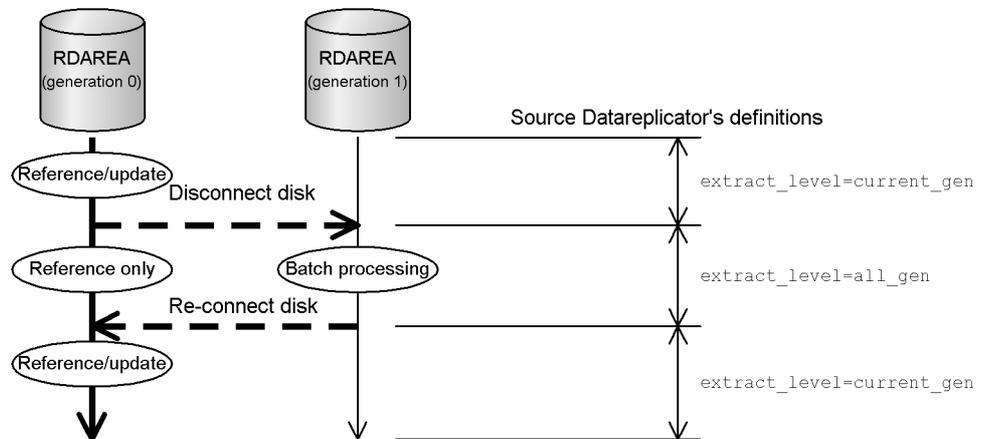
Some limitations apply to the specification of `extract_level=all_gen`. For details, see (2) *Limitations*.

The following are guidelines for specifying the `extract_level` operand:

Example: The current RDAREA is always the master (source subject to extraction)



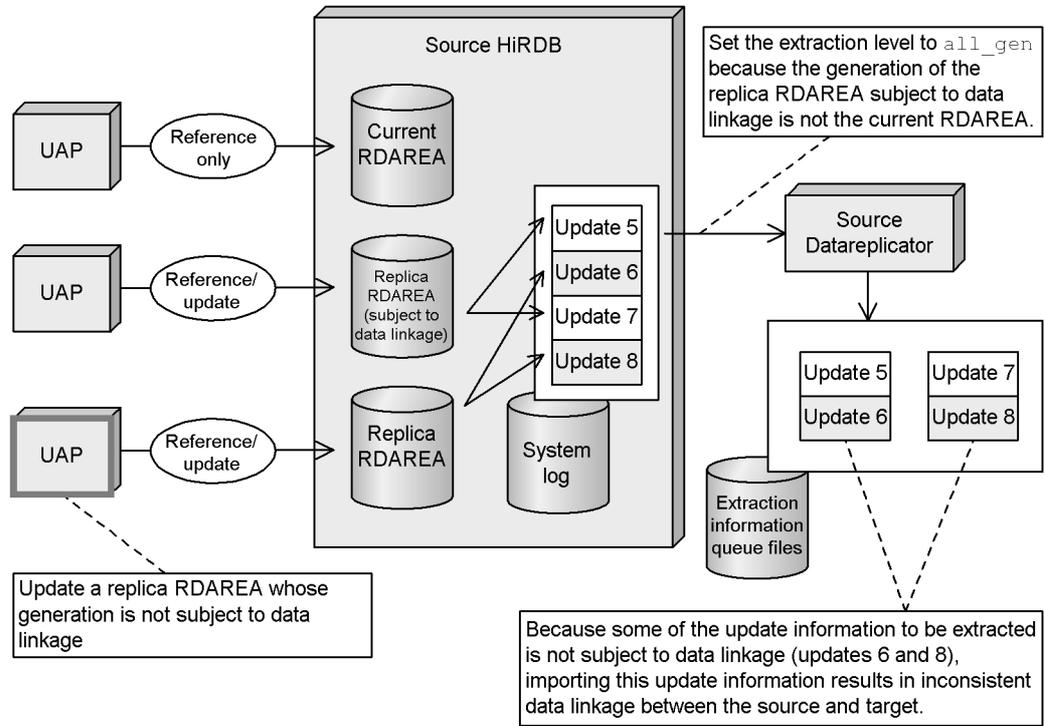
Example: An RDAREA other than the current RDAREA can be the master (source subject to extraction)



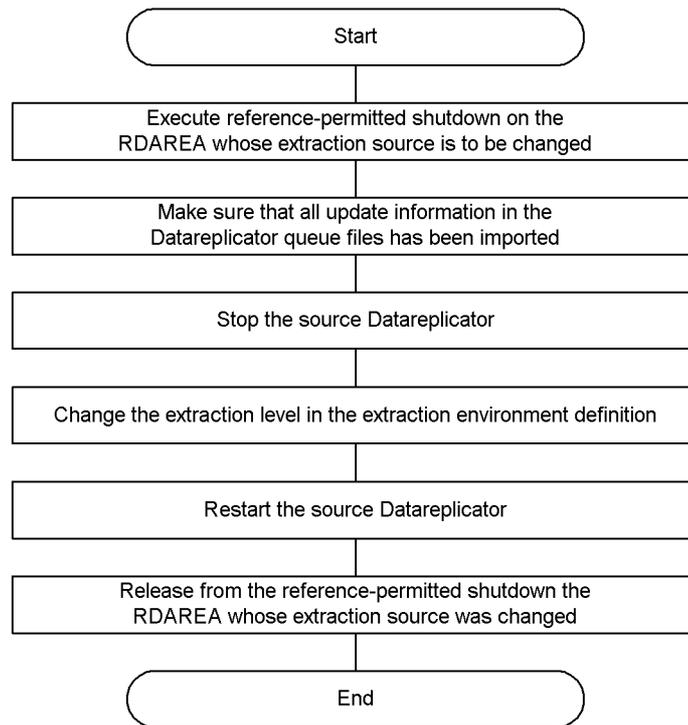
(2) Limitations

- For updating of the event table (event issuance), the update information for all generations is extracted, regardless of the source Datareplicator's definitions.
- For PURGE TABLE, if `extract_level=current_gen` is specified, the Datareplicator extracts only PURGE TABLE whose RDAREAs are all current RDAREAs.
- For the RDAREA subject to data linkage, specify `current_gen` as the extraction level for the current RDAREA if possible.

To use the replica RDAREA subject to data linkage as a non-current RDAREA, specify `all_gen` as the extraction level. In this case, updates on all replica RDAREAs are extracted regardless of whether they are subject to data linkage; therefore, updating a replica RDAREA that is not subject to data linkage results in inconsistent data linkage. When you set the extraction level to `all_gen`, make sure that only the replica RDAREAs subject to data linkage are updated. The following example shows inconsistent data linkage that occurs when the extraction level is set to `all_gen`:



- To change the source replica RDAREA that is to be extracted, the replica RDAREA must be synchronized with the target database. The following shows the procedure for changing the replica RDAREA subject to extraction:



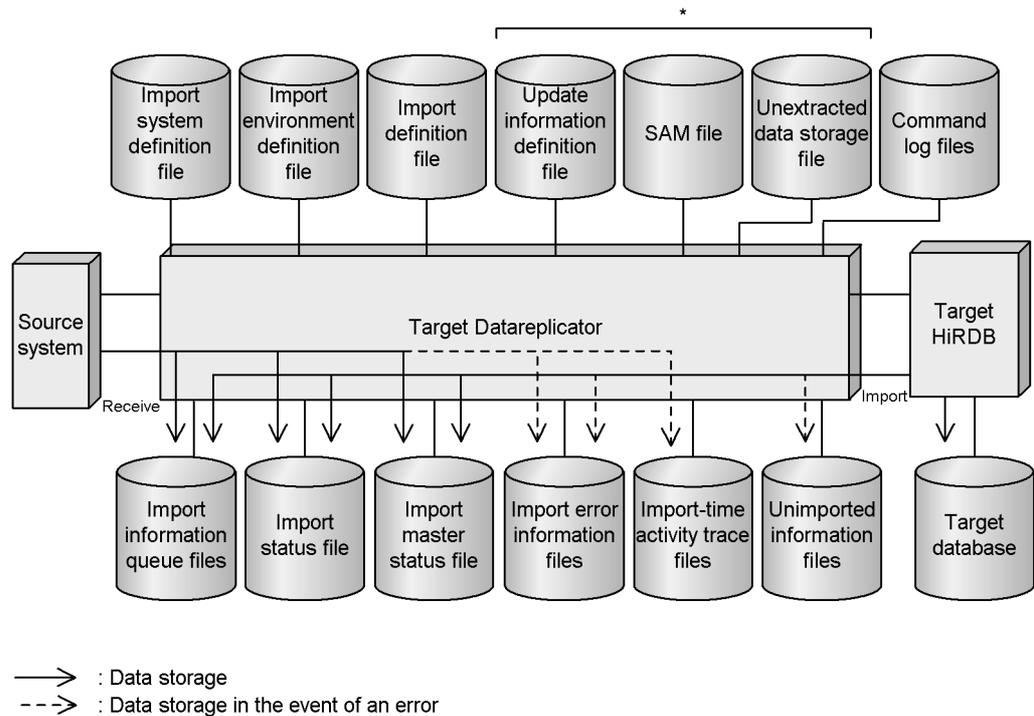
3.3 Target Datareplicator's import processing

This section explains the target Datareplicator's import processing. If the target database is a mainframe database (XDM/RD E2), see the *VOS3 XDM/DS* manual.

3.3.1 Overview of import processing

Import processing consists of *reception* and *import*. The following figure provides an overview of import processing.

Figure 3-20: Overview of import processing



* These files are required in order to link data with a mainframe database (PDM2 E2 or RDB1 E2) using SAM files, except that the update information definition file is not required for RDB1 E2.

(1) Reception

The target Datareplicator receives extraction definition and update information from the source system and stores this information in an *import information queue file*; it stores the reception status in the *import status file*.

If an error occurs during reception processing, Datareplicator outputs error

information to an *error information file*.

(2) Import

The target Datareplicator reads the update information in the import information queue file, issues an SQL statement for the row corresponding to the mapping key to update the target database, and stores the import status in the import status file.

If an error occurs during import processing, Datareplicator outputs the error information to an *error information file*. If an SQL statement results in an error at the HiRDB, Datareplicator outputs the SQL statement to an *unimported information file*.

3.3.2 Files and processes used during import processing

This section explains the files and processes that are used during import processing. For details about the files and processes used when JP1/Cm2 is used for operations management, see *3.4.5 Files and processes used for operations management*.

(1) Files used during import processing

The files discussed as follows are used during import processing.

(a) Import information queue files

The target Datareplicator uses these files to store the extraction definition and update information that is sent from the source system.

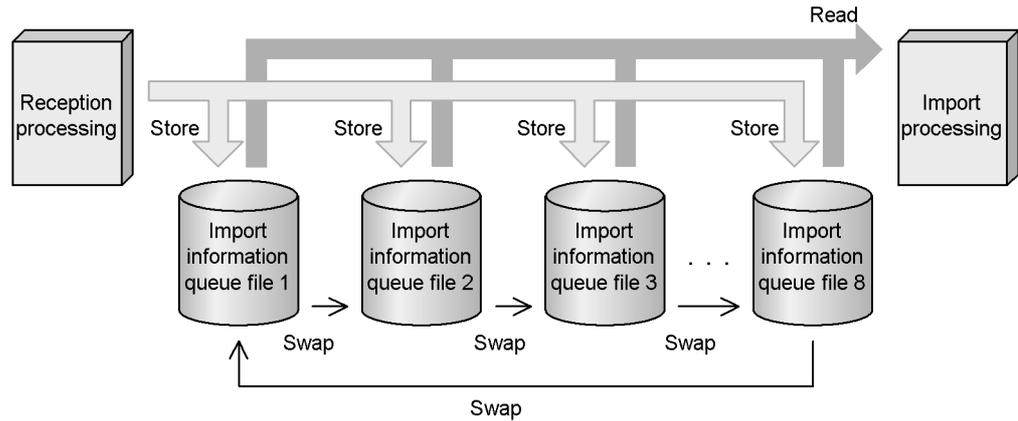
Datareplicator stores the received update information sequentially in one of the import information queue files. When this import information queue file becomes full, Datareplicator uses another import information queue file. This is called *swapping*, and it enables the target Datareplicator to store a large amount of update information and to import a large amount of information in the batch mode.

When all import information queue files become full, Datareplicator re-uses the first file. However, if import of the update information has not been completed for the file that is to be used next, swapping cannot take place. In this case, communication with the source system is broken and update information storage processing is suspended until the next transmission interval. If all update information in the file to be used during import processing in the next transmission interval has been read, Datareplicator resumes storing update information.

For details about the transmission interval, see *4.6.4 Designing the transmission procedure*. For details about the transmission interval specified with XDM/DS, see the *VOS3 XDM/DS* manual.

The following figure shows the procedure for storing data in the import information queue files.

Figure 3-21: Procedure for storing data in the import information queue files

**(b) Import-time status files**

These files store information such as the reception/import status required for recovery in the event of an error and the extraction definition status in the source system.

(c) Import master status file

This file stores the execution results during initial startup.

(d) Import-time error information files

If reception or import processing results in an error, Datareplicator outputs error details to an error information file. Datareplicator can also output the same information to the syslog file. The `syslogout` operand in the import system definition specifies whether error information is to be output to the syslog file.

(e) Import-time activity trace files

Datareplicator uses the activity trace files to collect Datareplicator's activity status. These files contain information about Datareplicator's operation and performance. The import-time activity trace files include the import trace files.

To use the activity trace files, specify the `int_trc_lvl` and `int_trc_filesz` operands in the import system definition. You can edit the obtained activity trace files with the `hdstrcredit` command. For details about the `hdstrcredit` command, see its command syntax in Chapter 7. *Command Syntax*.

(f) Unimported information files

An SQL statement issued for import processing that results in an error is output to an unimported information file. After import processing is completed, you can check the unimported information files, and then re-execute affected SQL statements and recover the errors.

Datareplicator creates two unimported information files per source system. When the one in use becomes full, Datareplicator starts using the other one. This is called swapping. When swapping occurs, Datareplicator starts output of SQL statements from the beginning of the file, regardless of the file status.

(g) Import system definition file

This file defines the target Datareplicator's overall operating environment, such as the method for establishing connection with the source system and the target HiRDB's access method.

(h) Import environment definition files

These files define operating environments for import processing, such as the names of the import information queue files and the import status files.

(i) Import definition files

These files define detailed information about import processing, such as the correspondence between the table and columns subject to extraction and the table and columns subject to import and the import groups.

(j) Command log files

These files store a record of the dates and times Datareplicator's commands are executed.

(k) Update information definition file

When the source database to be linked using a SAM file is PDM2 E2, the update information definition file defines information about import processing, such as the table and columns subject to extraction, column information, and column redefinition information in the SAM file.

(l) SAM file

When a SAM file is used for data linkage, this is the PDM2 E2's or RDB1 E2's update information extraction SAM file transferred by the file transfer program.

In the case of PDM2 E2, you use the statistical activity analysis utility (PDMJANL) to create SAM files. When you execute this utility, you must either specify `EXRANGE=SWAP` or omit `EXRANGE` in the `EXTRACT` statement. If you execute the utility with any other specification, termination information might not be output to the SAM file, in which case data linkage will fail. In the case of RDB1 E2, you use the update information extraction facility to create an information extraction SAM file. Use a variable-length blocked dataset for the SAM file (fixed-length blocked datasets are not supported for data linkage).

For details about PDM2 E2's statistical activity analysis utility (PDMJANL), see the *VOS3 PDM2, PDM2 E2 Utility* manual or the *VOS1 PDM2 E2 Utility* manual.

(m) Unextracted data storage file

When a SAM file is used to link data with the source database, Datareplicator uses the unextracted data storage file for output of update data for a table that is not specified in the extraction statement in the update information definition file or for output of update data information that cannot be imported.

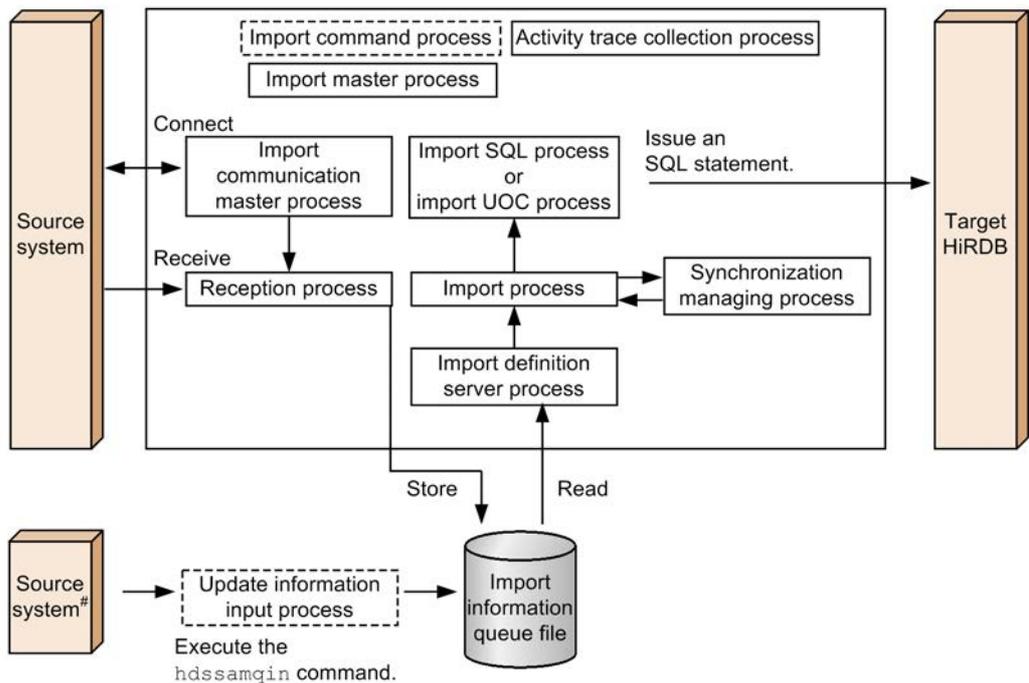
(n) import suppression list file

To use import suppress to skip errors that occur during import processing, create an import suppression list file. For details about using import suppression to skip errors, see 3.3.10(1) *Using import suppression to skip errors.*

(2) Organization of processes during import processing

The following figure shows the organization of processes during import processing.

Figure 3-22: Organization of processes during import processing



Legend:

- : Resident process
- : Nonresident process

#: Indicates a mainframe database using SAM files.

(a) Import command process

The import command process processes the target Datareplicator's command and issues an instruction to the import master process.

(b) Import master process

The import master process controls the entire target Datareplicator.

(c) Import communication master process

The import communication master process receives a connection request from the source system.

(d) Reception process

The reception process receives update information from the source system and stores it in an import information queue file.

(e) Import definition server process

The import definition server process controls the import process.

(f) Import process

The import process controls the import SQL process.

(g) Import SQL process

The import SQL process creates SQL statements on the basis of the update information and issues them to the target database.

(h) Import UOC process

The import UOC process issues SQL statements to the source database to process update information using a UOC routine.

(i) Update information input process

When a SAM file is used for data linkage, Datareplicator uses the update information input process to store the SAM file transferred from the mainframe database (PDM2 E2 or RDB1 E2) into the update information queue file using the `hdssamqin` command.

(j) Activity trace collection process

The activity trace collection process collects activity trace information.

(k) Synchronization managing process

The synchronization managing process manages global transactions when the import transaction synchronization facility is used.

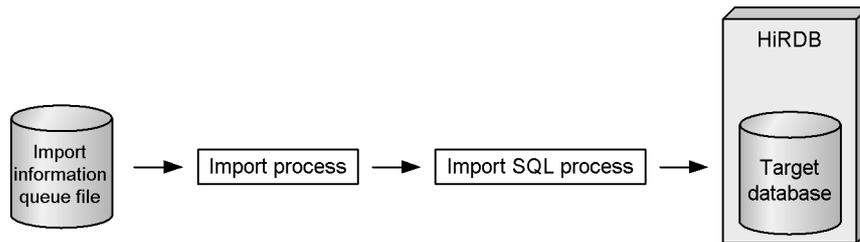
3.3.3 Import methods

The two import methods that are provided are the transaction-based import method and the table-based import method. The import environment definition determines the method that is to be used. For details about the system design, see 4.7.3(1) *Designing the import processing method*.

(1) Transaction-based import method

The import method that assigns one import process and one import SQL process is called the *transaction-based import method*. This method reads one transaction at a time from the import information queue file and imports them in order into HiRDB. The following figure shows the organization of processes for the transaction-based import method.

Figure 3-23: Organization of processes for the transaction-based import method



(2) Table-based import method

The import method that creates an import group for one or more tables subject to import processing and imports data for one group at a time is called the *table-based import method*.

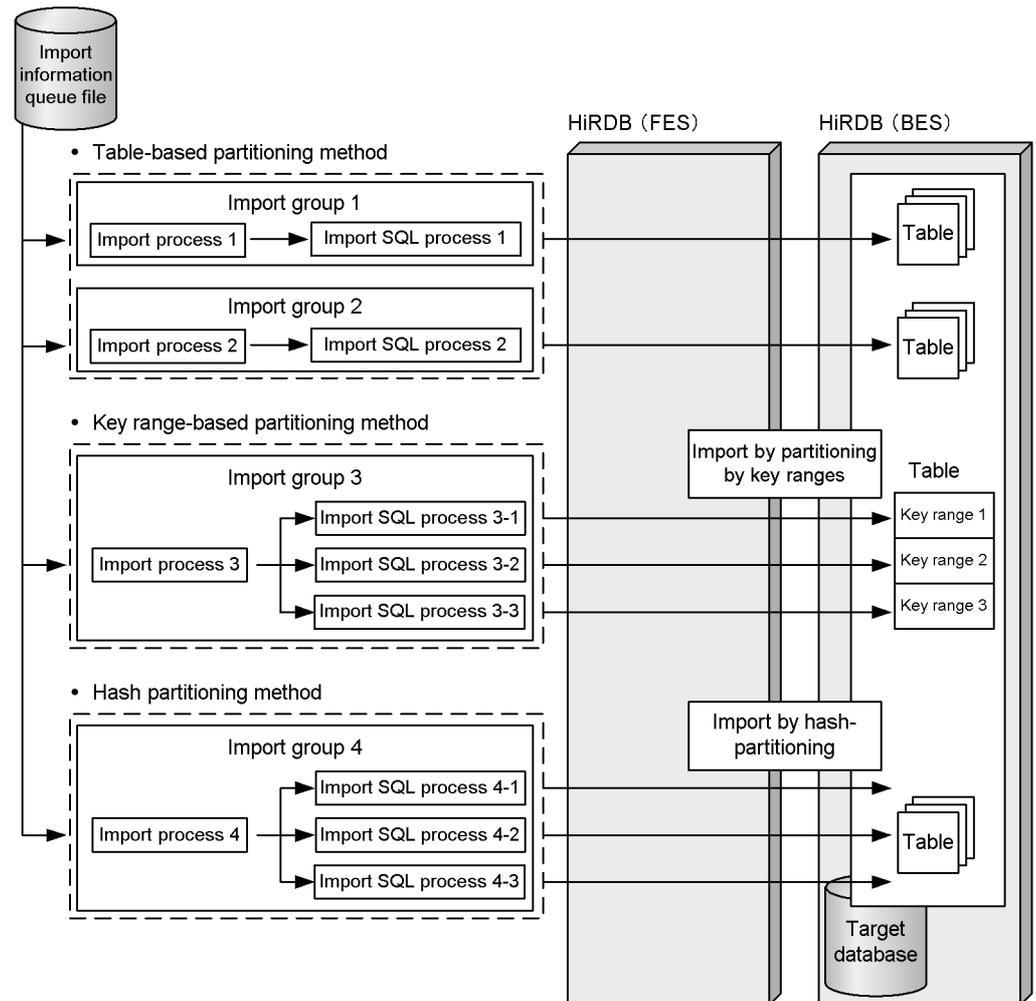
If the amount of update data is the same in each table subject to import processing, an improvement in import processing performance can be expected. However, the number of active processes increases, resulting in an increase in memory usage. When you use this import method, take into account the available memory capacity.

The table-based import method is broken down into the following types:

- Table-based partitioning method
- Key range-based partitioning method
- Hash partitioning method

The following figure shows the organization of processes for the table-based import method.

Figure 3-24: Organization of processes for the table-based import method



(a) Table-based partitioning method

The table-based partitioning method creates an import group for one or more tables that are subject to import processing. With this method, Datareplicator assigns one import process and one import SQL process to each import group. Each import process reads in order the update information in the import information queue file and issues SQL statements only to the tables assigned to the import group.

You can expect an improvement in throughput if the target HiRDB is a parallel server and if the tables are not row-partitioned among multiple servers.

(b) Key range-based partitioning method

The key range-based partitioning method creates one import group for one table that is subject to import processing and specifies key range partitioning conditions within the import group. With this method, Datareplicator assigns one import process and as many import SQL processes to the import group as there are key range partitions. The import process reads in order the update information in the import information queue file, and only the import SQL process that satisfies a specified condition issues an SQL statement.

You can expect an improvement in throughput when a target HiRDB is a parallel server, one table is row-partitioned among multiple units, and the key range partitioning conditions are defined by the target Datareplicator in the same manner as with the table's row-partitioning.

If data linkage does not require much workload for the target HiRDB, such as when there is only a small amount of data to be imported at one time or when the table subject to import processing is not large, you might be able to improve the performance of import processing by not using the key range partitioning method because range checking can be skipped.

If HiRDB is configured as described below, use of the key range-based partitioning method might not improve performance because of an increase in the communications load between front-end server and back-end server.

- The front-end server and back-end server are located at different machines.
- The HiRDB table is hash-partitioned.

Additionally, because the conditions for key range partitioning can be either match or range specifications, if consecutive values are sent to a column used as the partitioning key, processing becomes concentrated at the specified front-end server.

If key range partitioning does not improve performance for these reasons, use the hash partitioning method.

(c) Hash partitioning method

The hash partitioning method creates one import group for one table that is subject to import processing. With this method, Datareplicator assigns one import process and as many import SQL processes to the import group as there are hash partitions. The import process reads in order the update information in the import information queue file, and only the import SQL process that satisfies a specified condition issues an SQL statement.

If key range partitioning does not improve performance, use the hash partitioning method. The hash partitioning method can partition consecutive values that are concentrated at one location when the key range partitioning method is used. The hash partitioning method is especially effective when data is imported into a HiRDB/Parallel Server that uses the multi-FES facility.

3.3.4 Import method for the multi-FES facility

HiRDB's multi-FES facility is used to distribute the workload and improve throughput by defining multiple front-end servers; it is appropriate in a HiRDB/Parallel Server when throughput is poor because SQL processing has become concentrated at a single front-end server.

When you use the table-based import method, the target Datareplicator can improve throughput by supporting the multi-FES facility. With the table-based partitioning method, you assign a target front-end server for each import group; with the key range-based partitioning method, you assign a target front-end server for each key range. For details about the definitions required to implement the multi-FES facility, see *5.10 Import definition*.

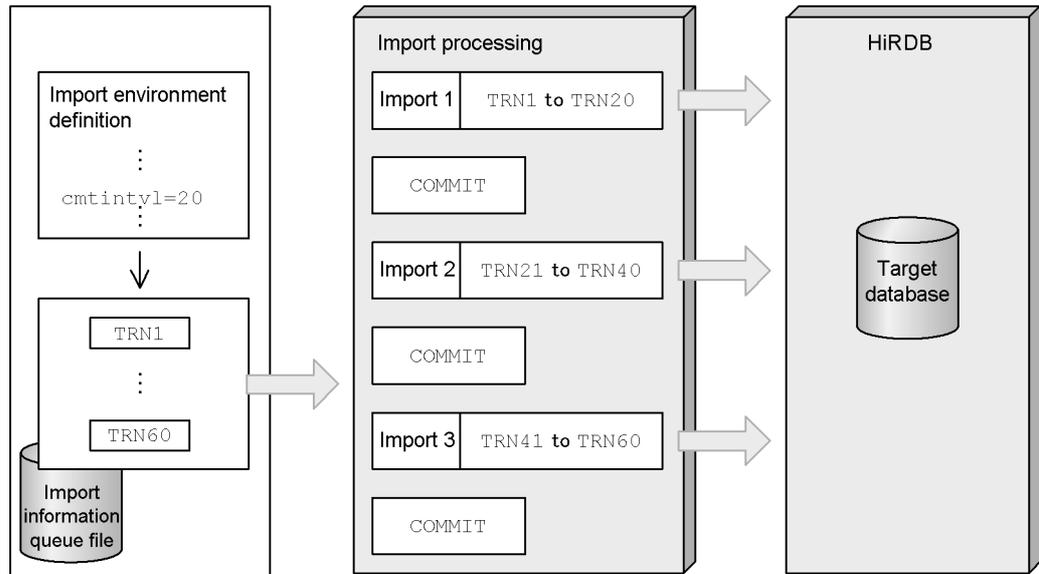
How to improve the efficiency of communication between HiRDB servers using the hash partitioning method

The hash partitioning method enables you to set the front-end server to be used to process import data for each partition. This helps to distribute the workload among the front-end servers, and if the front-end server and the back-end server containing the RDAREA are on the same server, it can also reduce the overhead associated with communication between the front-end server and back-end server.

3.3.5 Units of import processing

The target Datareplicator can import as a unit multiple transactions executed by the source system, which can improve performance. You use the `cmtintvl`, `trncmtintvl`, and `tblcmtintvl` operands in the import environment definition to specify the units of import processing. The following figure shows the units of import processing.

Figure 3-25: Units of import processing



3.3.6 Data linkage for a table with a trigger set

Two types of triggers can be defined for tables, one that is defined by the user, and one that is defined automatically by HiRDB. This subsection explains the tasks required for linking data in tables with triggers defined.

(1) Linking data in tables with triggers defined by the user

If data linkage is applied to a table with a trigger defined by the user, inconsistency might occur between the databases because the trigger SQL is executed on the source database, and then after data linkage, the trigger is executed again on the target database.

Such a database inconsistency might occur in the following cases:

- A BEFORE trigger uses an assignment statement of a routine control SQL.
- An AFTER trigger is defined for the target table that is to be replicated.

If either of these conditions is satisfied, specify `not_execute` in the `control_trigger` operand in the import environment definition.

Note:

- If you define a file output statement of a routine control SQL statement in the AFTER trigger for the target table and specify `not_execute` in the `control_trigger` operand, data will not be output to the file.

- If a trigger is defined for the target table and HiRDB is accessed from within a function of the import information editing interface, the `control_trigger` operand is ignored in the import environment definition.

(2) Linking data in tables with triggers defined by HiRDB

If `CASCADE` is specified as a referential constraint action for a table with a referential constraint defined, a trigger is defined for that table by HiRDB. If you will be applying data linkage to a table for which `CASCADE` is specified, make sure that the following conditions are satisfied:

Prerequisites

- In the import environment definition, specify `not_execute` in the `control_reference_trigger` operand.
- In the import environment definition, specify `use` in the `check_pending` operand.
- Apply data linkage to all tables for which a referential constraint is defined.
- The constraints for referenced tables and referencing tables are the same between the source and the target.

Limitations

A table for which a referential constraint is defined cannot be specified in an import information editing UOC routine.

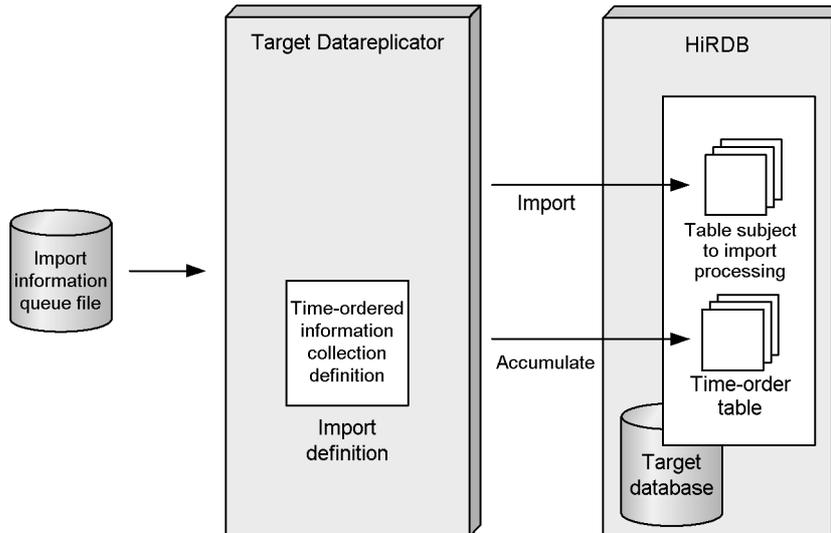
If all these conditions are satisfied and a referential constraint is defined for tables that are subject to import processing, the corresponding tables are placed in check pending status. Execute the integrity check utility for each table and release the tables from check pending status.

3.3.7 Collecting time-ordered information

If you specify collection of time-ordered information in the import definition, you can accumulate the update information for the target system in the time-ordered information table during import processing. By collecting update information in chronological order, you can retrieve a history of updates as an update ledger or analyze the frequency of update processing. In addition to update information, you can also collect information such as the import dates, import times, and update types. For details about the available types of time-ordered information, see *4.3.7 Creating a time-ordered information table*.

The following figure shows the concept of collecting time-ordered information.

Figure 3-26: Concept of collecting time-ordered information



3.3.8 Collecting information about the target Datareplicator

Commands and system definitions enable you to obtain the information about the target Datareplicator that is explained as follows.

(1) Status information

By executing the status information collection command (`hdsstate` command), you can output *status information* to the standard output. This information can be used to check the target Datareplicator's status. You can collect status information only while the target Datareplicator is active.

The status information includes the utilization status of the import information queue files, reception processing information, and import processing information. For details about how to collect status information, see the `hdsstate` command in Chapter 7.
Command Syntax.

(2) Activity trace

You can collect an *activity trace* to obtain Datareplicator's activity status and to determine its performance. You can use an activity trace for the following purposes:

- If import processing seems slow, use the activity trace for guidance in changing system settings to maximize HiRDB performance (for example, for determining whether tuning should involve using indexes for the HiRDB table definitions, or the concurrency level could be improved by executing Datareplicator's import processing in the table-based mode).

- If an import processing editing UOC routine's performance is poor, use the activity trace to determine whether the problem is with Datareplicator or with the UOC routine.

To obtain an activity trace with the target Datareplicator, you specify the `int_trc_lvl` and `int_trc_filesz` operands in the import system definition. To obtain an activity trace for each data linkage identifier, you specify the `int_trc_getl` operand in the import environment definition. Use the `hdstrcredit` command to output an obtained activity trace.

For details about activity trace definitions, see *5.8 Import system definition* and *5.9 Import environment definition*; for details about the `hdstrcredit` command, see the `hdstrcredit` command in Chapter 7. *Command Syntax*.

3.3.9 Processing update information with a user own coding routine

To aid data linkage applications, you can create the following user own coding (UOC) routines for the target Datareplicator:

- Import information editing UOC routine
This UOC routine enables you to edit update information.
- Column data editing UOC routine
This UOC routine enables you to edit update information for any column.

For details about UOC routines, see Chapter 8. *User Own Coding Routines*.

3.3.10 Skipping import errors

If an error occurs during import processing, Datareplicator enables you to ignore the error and continue the import processing. This prevents interruption of import processing when errors that do not affect the import processing occur.

There are two types of error skipping during import processing:

Type	Description
Using import suppression to skip errors (skipping specified errors)	After an error occurs, this method specifies whether the error is to be skipped on the basis of the error information (update information identifiers). This method is applicable to all errors, including SQL errors.
Skipping SQL errors	This method uses <code>SQLCODE</code> to specify in advance the SQL errors that are to be skipped. This method is applicable only to some SQL errors. For details about the applicable SQL errors, see the <code>skip_sqlcode</code> operand in <i>5.9 Import environment definition</i> .

(1) Using import suppression to skip errors

To ignore an error and continue import processing, specify the *update information identifier*[#] that is displayed as `information` in the error message. Use the import suppression list file to specify the update information identifiers.

For details about the message, see *10.1.3(4) Explanation of detail information*.

#

You can also obtain the update information identifier by executing the `hdsrefinfmt` command. For details, see *hdsrefinfmt* in Chapter 7. *Command Syntax*.

(a) Prerequisites

The following table shows the Datareplicator versions that are capable of using import suppression to skip errors.

Table 3-1: Datareplicators capable of using import suppression to skip errors

Extraction product	Version
HiRDB Datareplicator	04-00-/O or later
HiRDB Datareplicator Version 5.0	05-03-/D or later
HiRDB Datareplicator Version 6	06-01-/A or later
HiRDB Datareplicator Version 7 or later	07-00 or later
XDM/DS (TMS/4V)	07-01 or later
XDM/DS (other than TMS/4V)	08-03 or later
VOS3 Database Datareplicator	01-00 or later

Note

If you use the following system configuration, even a Datareplicator of the applicable version listed in Table 3-1 cannot use import suppress to skip errors:

- The source system uses HiRDB Datareplicator Extension.
- The `hdssamqin` command is used to input data to the target Datareplicator.

(b) Definition method

To specify use of import suppress, use an import suppression list file. The following table provides the details of an import suppression list file.

Table 3-2: Details of import suppression list file

Item	Description
File name	HDSPATH/reflect_pass_list_data-linkage-identifier# Example: HDSPATH/reflect_pass_list_a1
File type	UNIX regular file or Windows file
Attributes	The user who executes the hdsstart command has the read privilege.

#

- Specify the linkage identifier in lower-case letters.
- For linkage identifiers 00 to 09, specify 0 to 9.
For linkage identifiers 0a to 0f, specify a to f.

The following shows the specification format in the import suppression list file:

Specification format

```
control-code = Trn[:Pos][,Grp] #comment
:
```

control-code

Specifies one of the following control codes, as appropriate to the operation:

Control code	Description
SKIP_TYPE_ONLY	Specifies normal import suppress. This code suppresses the specified update information identifier.
SKIP_TYPE_UNTIL	Specifies data linkage recovery.

```
Trn[:Pos][,Grp]
```

Specifies the update information identifier that is displayed as information in the error message. There can be no spaces or tabs between Trn, Pos, and Grp.

Specification	Description
Trn	ID of the transaction whose update information has been stored. This is the hexadecimal value (24 digits) preceding the colon in Extract Id in the information portion of the error message.

Specification	Description
Pos	Storage position of the update information in the transaction indicated by Trn. This is the decimal value (1 to 4294967295) following the colon in Extract Id in the information portion of the error message.
Grp	Import group name specified in the group statement in the import definition. The error to be suppressed by Grp specification depends on the method of import processing. Table 3-3 describes Grp suppression for each import method.

Table 3-3: Grp suppression for each import method

Import method	Error to be suppressed by Grp specification	
	Grp specified	Grp not specified
Table-based import method	Suppresses Trn:Pos in the import group specified in Grp.	Suppresses Trn:Pos in all import groups.
Transaction-based import method	Suppresses Trn:Pos in all import groups (Grp specification is ignored).	Suppresses Trn:Pos in all import groups.

Rules

- Specify only one item per line.
- Specify information from one control code through update identifier on the same line.
- One line cannot exceed 512 bytes.
- If a line contains a hash mark (#), the information following the hash mark is treated as a comment.

Example

This subsection presents an example of specifying an update information identifier.

This example assumes that the following message has been displayed during import processing by the table-based import method:

information portion of the message

```
Fri May 23 16:29:42 2003 process: hdssqle[trngroup](2032)
function: hds_sqe_hexeis
errorcode: KFRB03033-E 02 a1 (hdssqle.exe[trngroup])
HiRDB data base EXECUTE error occurred, table name =
"k896201"."R003", SQLCODE = -803.
information: "C1 "=1
KFPAl1803-E Duplicate key value in unique index id=196861
```

```
Additional Transaction Info = 3e681b950000000000000012f.
Extract Id = 3e681b95000000000000000012f:1
```

To suppress import of only the transaction 3e681b950000000000000012f (storage position: 1) for the import group `trngroup`, this example is specified as follows:

Contents of the import suppression list file

```
SKIP_TYPE_ONLY=3e681b95000000000000000012f:1, trngroup
```

(2) *Skipping SQL errors*

If an error with a predefined `SQLCODE` occurs, Datareplicator can ignore the error and continue import processing. With this method, you can skip only SQL errors, but it is easier than using import suppress because you can specify in advance the errors that are to be skipped.

If an `SQLCODE` error for which `skip` is specified occurs during import processing, you can ignore the error and continue import processing. You must specify the applicable `SQLCODE`s in the `skip_sqlcode` operand in the import environment definition. Note that Datareplicator ignores the `skip` specification for an `SQLCODE` corresponding to an SQL error that is subject to implicit rollback; in this case, import processing terminates.

For details about the `skip` specification for import SQL errors, see the `skip_sqlcode` operand in *5.9 Import environment definition*.

3.3.11 Creating a merge table

An error results at the target system if you attempt to import update information extracted from multiple tables at the source system into a single table at the target (importing n tables into 1 table) or if you attempt to import data when the source system's database is not a structured or relational database. Such an error occurs because of duplication of key values or because there are no corresponding rows according to the mapping key. To resolve these limitations, the target Datareplicator enables you to convert `INSERT` to `UPDATE` when a key subject to `INSERT` is duplicated in the update information or to convert `UPDATE` to `INSERT` when there is no corresponding `UPDATE` row. This is called creation of a *merge table*.

You specify merge table creation in the import table definition. For details about how to specify an import table definition, see the `load` statement in *5.10 Import definition*.

(1) *Prerequisites for a merge table*

A merge table supported for import processing must satisfy the following conditions:

- It must be a general table, not a time-ordered information table.

- It must be a non-FIX table. If it has the FIX attribute, the null value storage option is not specified with DELETE. This condition is not applicable if a merge table is not created from *n* tables.
- Import processing must use an SQL process. You cannot create a merge table for import processing if data has been edited by an import information editing UOC routine.
- The mapping keys must be the same (applicable to a merge table from *n* tables). There must also be a unique index of all or part of the mapping key, and there must be no unique index for non-mapping key columns.
- PURGE TABLE cannot be executed on the extracted table (applicable to a merge table from *n* tables).
- There can be no updating of the mapping key (applicable to a merge table from *n* tables).

(2) Merge table processing

A merge table is created according to the following rules:

- If UPDATE data is not found, it is stored as INSERT data.
- If INSERT data results in a key value duplication, the data is stored as UPDATE data.
- For data subject to DELETE, either the corresponding row is deleted or the null value is placed in the corresponding column, depending on an import definition option.

(3) Importing a merge table into a table containing a repetition column

If you execute UPDATE specifying an element number for a table containing a repetition column, only the updated element data is sent as the update information, not the entire element data in the corresponding column. In this case, Datareplicator compensates the table with the null value so that there will be enough data items for INSERT during merge table import processing. The following table shows this compensation method:

SQL		Element # < minimum element #	Minimum element # < element # < maximum element#	Maximum element # < element #
INSERT		Compensation not needed		
UPDATE SET	Column specified	Compensation not needed		
	Element specified	NULL compensated	NULL compensated	Not compensated

SQL	Element # < minimum element #	Minimum element # < maximum element #	Maximum element # < element #
UPDATE ADD	NULL compensated	Impossible	Not compensated
UPDATE DELETE	Imported as NULL column value		

Note

Element # indicates the element number. *Maximum element #* and *Minimum element #* indicate the maximum and minimum element number contained in the update information for the corresponding repetition column.

(4) Notes about the data that is imported into a merge table

- When concatenation operation data is imported into a merge table and there is no corresponding row for UPDATE (SQLCODE=100), only the concatenation operation data is imported by INSERT.
- Do not import into a merge table BLOB-type or BINARY-type data to which backward deletion updating is applied. Inconsistency might result between the source and target databases.

3.3.12 Specifying the synchronization point processing for import processing

When Datareplicator issues an import processing request to the target database, information in the queue file whose import processing is completed is written into a status file. This constitutes transaction processing. The processing that determines whether a transaction is to result in normal termination or an error (commit or rollback) is called *synchronization point processing*. Once synchronization point processing has been completed, the update information will remain unchanged even if import processing is restarted (rerun).

(1) Types of synchronization point processing methods for the target database

The synchronization point processing methods for the target database include the single-phase commit method and the two-phase commit method.

- Single-phase commit method
This method executes synchronization point processing with a single commit for the target database.
- Two-phase commit method
This method executes the commit request twice (at two phases) for the target database.

You use the `commitment_method` operand in the import system definition to specify

the synchronization point processing method. If you change the synchronization point processing method, you must execute an initial startup of the target Datareplicator.

(2) *Difference between single-phase and two-phase commit methods*

When a commit for the target database terminates normally, Datareplicator writes the queue file's information on the update information up to the commit point, and then starts the next transaction for processing update information. If an error occurs at the target database, Datareplicator terminates import processing, and then waits for a restart. When the single-phase commit method is used, Datareplicator re-executes the transaction that resulted in the error during the previous commit processing. In this case, Datareplicator might not be able to identify the completion of commit processing at the target database (such as when control is not returned normally to Datareplicator due to a communications error during control return from the target database). When this happens, Datareplicator assumes an error and re-executes the transaction when HiRDB is restarted (rerun). If the mapping key is not unique, duplicate data might be inserted.

The two-phase commit method can prevent such insertion of duplicate data. If your version of the target Datareplicator supports the two-phase commit method, use it (define it for use).

3.3.13 Suppressing message output

While it is operating, Datareplicator outputs to the syslog file (or event log for Windows), and to error information files messages that report the activity status. These messages are important to determine the activity status; however, many messages might be output depending on the operating environment and they might use up a large amount of resources.

To minimize this problem, you can suppress output of some messages that you determine to be unnecessary. The following table shows the operands used to suppress message output:

Operand	Output destination subject to suppression
syslogout	syslog file (or event log)
syslog_message_suppress	
info_message_out [#]	<ul style="list-style-type: none"> • syslog file (or event log) • Error information file

Note

For details about each operand, see *5.8 Import system definition*.

[#]: If you have specified `suppress` in the `info_message_out` operand to suppress output of specific messages, you can use the `except_suppress` operand to exclude

any of the specified messages so that those messages will be output to the console and error information files.

The following table shows the results of message output when multiple operands are specified to suppress message output to the syslog file (or event log):

info_message_out	Operand		Message output destination	
	syslogout	syslog_message_suppress	syslog file (or event log)	Error information file
suppress	Not specifiable	Not specifiable	N	N
nosuppress	true	Specified	N	Y
		Omitted	Y	Y
	false	Not specifiable	N	Y

Y: Output.

N: Not output.

3.3.14 Processing in the event of a lock error

If import processing results in a lock error, the target Datareplicator re-executes internally the transaction import processing that resulted in the error. If three attempts to re-execute the processing do not resolve the lock error, the import processing terminates with an error.

3.4 Using JP1/Cm2 for operations management

You can use NETM**Cm2's* include the corresponding JP1/OpenView products.) SNMP agent facility to manipulate a Datareplicator data linkage system at a remote machine.

3.4.1 Overview of using JP1/Cm2 for operations management

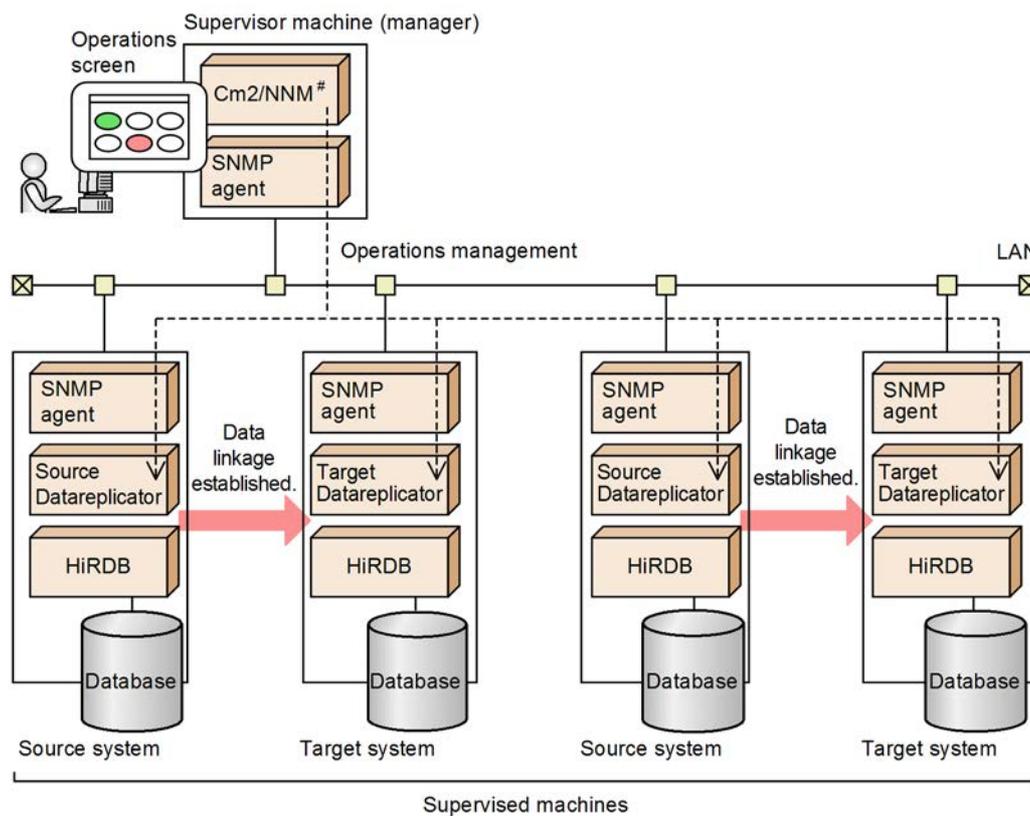
When many data linkage systems are being used, operations might be performed without an operator at each machine outside the computer center. If an error occurs in one Datareplicator in such a system configuration, the operator at the computer center might not be able to detect the system shutdown, resulting in a delay in taking necessary actions.

JP1/Cm2's SNMP agent facility makes it possible for multiple Datareplicator systems that do not have on-site operators to be managed remotely from the computer center. The system at the computer center (*manager*) that operates remote locations is called the *supervisor machine*, while a Datareplicator system that is subject to manipulation from the manager is called a *supervised machine* or a *machine subject to supervision*.

Datareplicator (for either UNIX or Windows) is the only data linkage product that can be managed by JP1/Cm2. JP1/Cm2's operations management is not applicable to the VOS3 and VOS1 data linkage products. For details about JP1/Cm2's SNMP agent facility, see the *JP1 Version 8 JP1/Cm2/Extensible SNMP Agent* manual and other related JP1/Cm2 manuals.

The following figure provides an overview of operations management when JP1/Cm2 is used.

Figure 3-27: Overview of operations management when JP1/Cm2 is used



#

Indicates JP1/Cm2/Network Node Manager.

(1) Operations management provided by JP1/Cm2

Using JP1/Cm2 for operations management provides the following facilities:

Status monitoring

This facility displays on the JP1/Cm2/Network Node Manager screen at the supervisor machine an icon for each Datareplicator; these icons change color depending on Datareplicator's activity status.

Information collection

This facility collects automatically information about Datareplicator at each machine that is using JP1/Cm2/Network Node Manager via the SNMP protocol. You can use this information for automatic monitoring of Datareplicator status and to issue a warning to the supervisor machine if Datareplicator terminates.

Remote control

This facility executes a remote Datareplicator's commands via the JP1/Cm2/Network Node Manager's SNMP protocol.

(2) Products required for use of JP1/Cm2 for operations management

To implement use of JP1/Cm2 for operations management, you must have the following products:

At the supervisor machine

JP1/Cm2/Network Node Manager

JP1/Cm2/SNMP System Observer

JP1/Cm2/Extensible SNMP Agent (UNIX)

At a supervised machine

JP1/Cm2/Extensible SNMP Agent (UNIX)

JP1/Cm2/Extensible Agent (Windows)

In the case of a supervisor or supervised machine that is in a Windows system, you must install the Windows SNMP service.

3.4.2 Status monitoring

On the JP1/Cm2/Network Node Manager's display screen at the supervisor machine, you must create icons for displaying each Datareplicator's activity status. The color of an icon at the supervisor machine tells you whether Datareplicator's source or target process is functioning normally.

If machines are grouped, the color of the applicable machine's and group's icon changes as its Datareplicator status changes. This makes it easy to identify an erroneous machine among many machines.

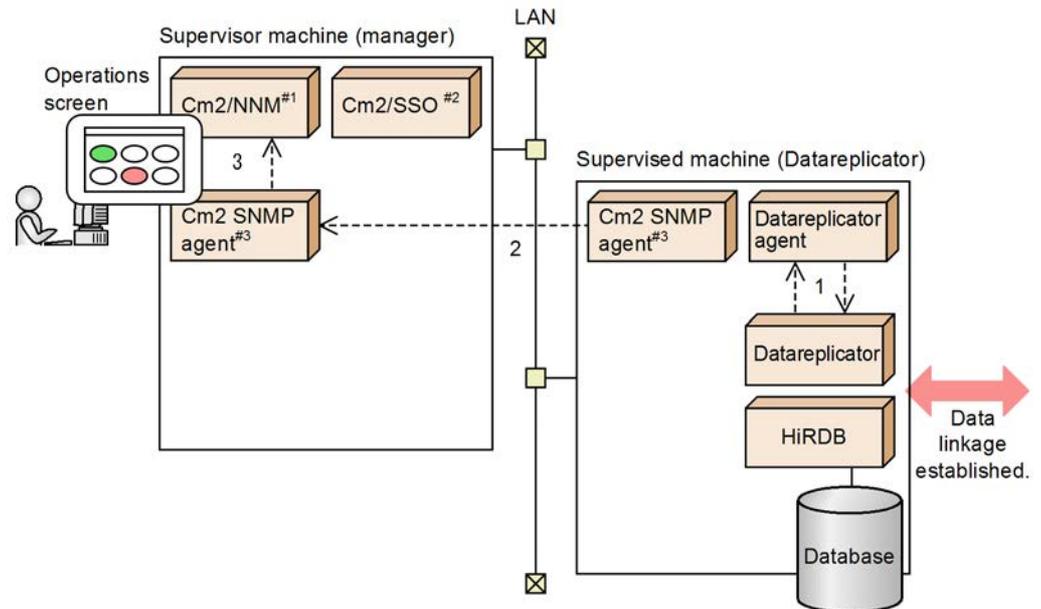
(1) Mechanism of status monitoring

To monitor a machine's status, you must install a Datareplicator agent at the subject machine. Datareplicator agent collects the status of all Datareplicators installed on the machine and passes the information to JP1/Cm2's SNMP agent. The SNMP agent passes this Datareplicator status information to the JP1/Cm2/Network Node Manager at the supervisor machine. Datareplicator's icon color on the JP1/Cm2/Network Node Manager screen is determined on the basis of this information.

(2) Icon colors displayed at the supervisor machine

Under the JP1/Cm2/Network Node Manager's default settings, green indicates normal, yellow indicates command shutdown, and red indicates error shutdown. The following figure provides an overview of using JP1/Cm2 for status monitoring.

Figure 3-28: Overview of using JP1/Cm2 for status monitoring



#1

JP1/Cm2/Network Node Manager

#2

JP1/Cm2/SNMP System Observer

#3

JP1/Cm2's SNMP agent

Explanation

1. At a specified interval, the supervised machine's Datareplicator agent collects information about each Datareplicator process. This monitoring interval can be changed by the `hdsagtopt` command.
2. Depending on the processing status, the supervised machine's Datareplicator agent sends information to the JP1/Cm2 SNMP agent.
3. The supervisor machine's JP1/Cm2 SNMP agent passes the information to JP1/Cm2/Network Node Manager and changes the icon color.

3.4.3 Information collection

You can use the JP1/Cm2/Network Node Manager's SNMP protocol to collect

information about Datareplicator at each machine.

The JP1/Cm2/Network Node Manager collects and accumulates data from each machine at a specified interval. If numeric data is collected, you can plot a graph to display trends in the number of extracted/imported data items over time. You can also define an event that will issue a warning when a monitored value reaches a specified value. Such information can be used as a basis for automatic monitoring of Datareplicator's status and for issuing a warning if Datareplicator terminates. For details about the Cm2/Network Node Manager's operation, see the manual *JP1 Version 8 JP1/Consolidated Management 2/Network Node Manager*.

MIB file

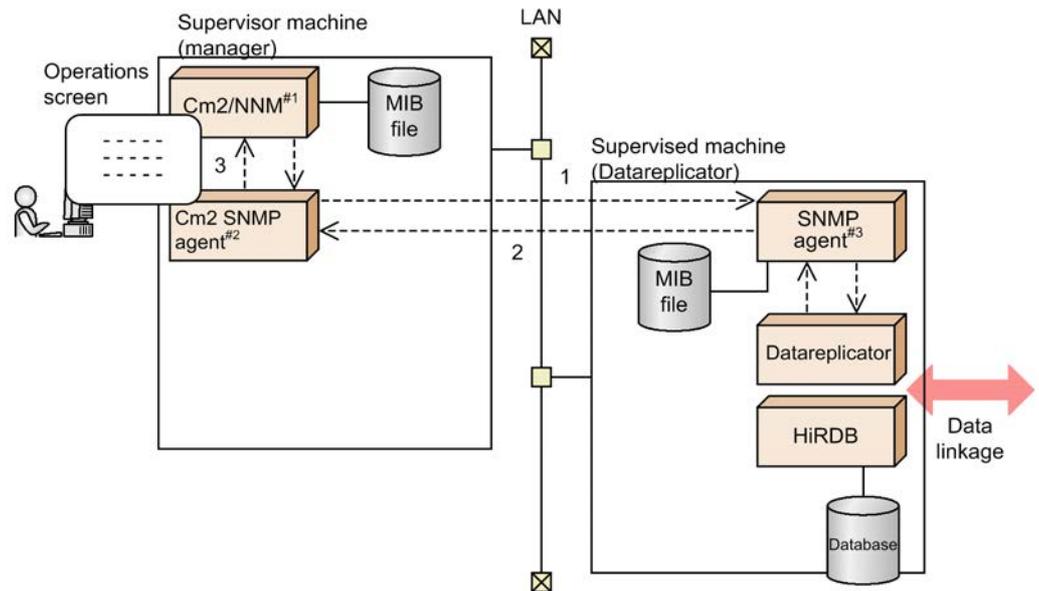
There must be a MIB file at each Datareplicator (at the supervisor machine and at each supervised machine) to collect information. Copy a Datareplicator MIB file (the filename is `hdsMIB`) into the supervisor machine, and then load it with the JP1/Cm2/Network Node Manager. Do not edit a Datareplicator MIB file. For details about the MIB file, see *3.4.9 MIB file*.

(1) Mechanism of information collection

The JP1/Cm2/Network Node Manager references the MIB file and issues an object ID and a `GET`, `GETNEXT`, or `SET` request to the supervised machine's SNMP agent. The supervised machine's SNMP agent references the MIB file to return the information that has been collected to the JP1/Cm2/Network Node Manager. When there are multiple Datareplicators in a HiRDB/Parallel Server configuration, information is collected for each machine.

The following figure provides an overview of using JP1/Cm2 for information collection.

Figure 3-29: Overview of using JP1/Cm2 for information collection



#1

JP1/Cm2/Network Node Manager

#2

JP1/Cm2's SNMP agent

#3

UNIX: JP1/Cm2's SNMP agent

Windows: Datareplicator's SNMP agent

Explanation

1. The supervisor machine's JP1/Cm2/Network Node Manager references the Datareplicator's MIB file (`hdsmib`) and passes the object ID and GET, GETNEXT, or SET request code to the supervised machine's SNMP agent.
2. The supervised machine's SNMP agent references the Datareplicator's MIB file (`hdsmib`) and collects Datareplicator information, and then passes the collected information to the supervisor machine's SNMP agent.
3. The supervisor machine's SNMP agent passes the information to the JP1/Cm2/Network Node Manager.

3.4.4 Remote control

You can use the JP1/Cm2/Network Node Manager's SNMP protocol to execute a Datareplicator's commands at a remote machine.

MIB file

There must be a MIB file at each Datareplicator (at the supervisor machine and at each supervised machine) to perform remote control. Copy a Datareplicator MIB file (the filename is `hdsMIB`) into the supervisor machine, and then load it with the JP1/Cm2/Network Node Manager. Do not edit a Datareplicator's MIB file. For details about the MIB file, see *3.4.9 MIB file*.

(1) Mechanism of remote control

You must define a command and its options in the JP1/Cm2/Network Node Manager at the supervisor machine. The JP1/Cm2/Network Node Manager issues a `SET` request and `SET` information to the supervised machine's SNMP agent on the basis of information in the MIB file. The supervised machine's SNMP agent references the MIB file and executes Datareplicator commands. However, the command execution results (return value) are not returned.

Details about a command executed by remote control are output to the supervised machine's system log and event log, except for the percent sign (%). When a command is executed successfully, it is logged into the command log in the directory.

(2) Limitations on available commands

Remote control enables you to execute only Datareplicator commands; no other commands can be executed under remote control. No checking is performed to validate the correctness of the command name or options. Datareplicator will attempt to execute any specified command, even if it is not a valid command.

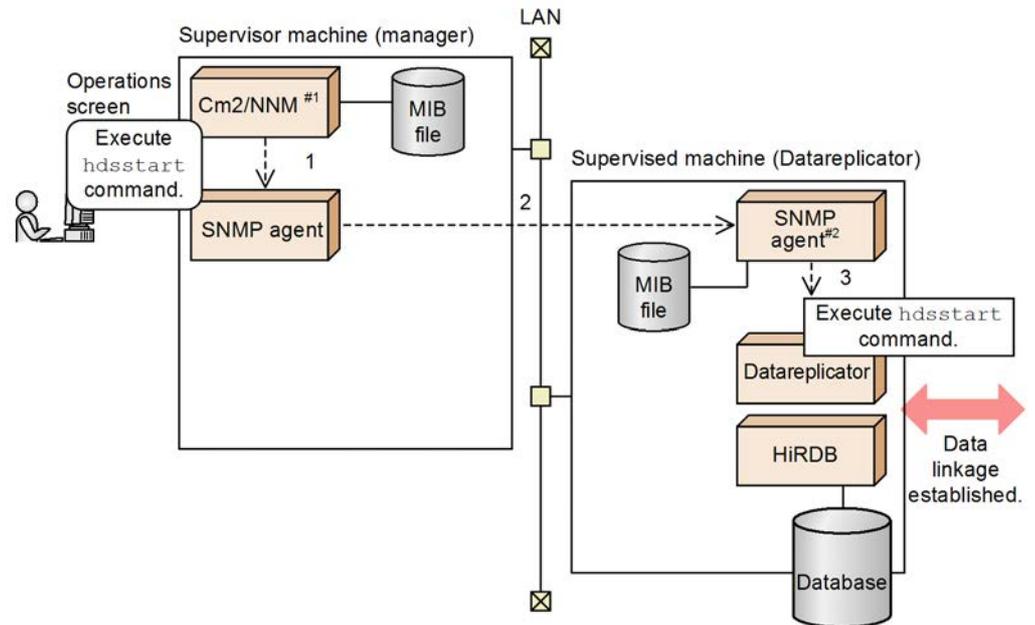
Although the SNMP agent can execute commands, it cannot receive a command's response. For this reason, you cannot execute an interactive command (one that issues a query message after it executes).

Unsupported command

```
hdestart -i command
```

The following figure provides an overview of using JP1/Cm2 for remote control.

Figure 3-30: Overview of using JP1/Cm2 for remote control



#1

JP1/Cm2/Network Node Manager

#2

UNIX: JP1/Cm2's SNMP agent

Windows: Datareplicator's SNMP agent

Explanation

1. Specify a Datareplicator command and options on the JP1/Cm2/Network Node Manager at the supervisor machine. The JP1/Cm2/Network Node Manager references the Datareplicator's MIB file and passes the command name, options, SET request code, and system index to the supervised machine's SNMP agent.
2. The supervised machine's SNMP agent references the Datareplicator's MIB file and executes the specified command and options.
3. The Datareplicator command is executed. Because the command execution results are not returned to the JP1/Cm2/Network Node Manager, you must use status monitoring or the information collection facility to check the results.

3.4.5 Files and processes used for operations management

This section explains the files and processes that are used by Datareplicator during JP1/Cm2 operations management. The files and processes described here are common to both source and target Datareplicators.

(1) Files used for operations management

Datareplicator uses the following type of file during JP1/Cm2 operations management:

MIB file (hdsMIB)

A MIB file contains a Datareplicator's information and object ID. You copy a MIB file into the supervisor machine's JP1/Cm2/Network Node Manager and register it. You must not edit a Datareplicator's MIB file.

(2) Organization of processes for operations management

The processes discussed as follows are used during JP1/Cm2 operations management.

(a) Datareplicator agent process

This process monitors the Datareplicator status. Once started by the `hdsagtstart` command, the Datareplicator agent process is made resident in the system. The agent process collects Datareplicator information at a specified interval.

(b) Information collection process

This process collects Datareplicator's activity information. The Datareplicator's agent process, the SNMP agent process, and the JP1/Cm2's SNMP agent start up to perform the process. When information is returned to the initiator process, the information collection process terminates.

(c) Datareplicator SNMP agent process (applicable to Windows Datareplicator only)

This process runs only under Windows SNMP service. It starts the information collection process to collect Datareplicator information, and then returns the collected information to the SNMP service.

3.4.6 Initializing operations management

This subsection discusses the items that you define to use JP1/Cm2 operations management.

(1) Setting up the supervisor machine

The item discussed as follows must be set up at the supervisor machine (manager).

(a) Loading the MIB file

Copy the Datareplicator MIB file (`hdsMIB`) into any directory in the supervisor machine. The information collection and remote control facilities become available

once you load the JP1/Cm2/Network Node Manager into the MIB. The following are the names of the directory and file where you will find the Datareplicator MIB file:

Directory name

For UNIX Datareplicator: `/opt/hirdbds/lib`

For Windows Datareplicator: `Datareplicator-installation-directory\lib`

Filename

`hdsMIB`

(2) **Setting up a supervised machine (for a UNIX Datareplicator)**

In the case of a UNIX Datareplicator, the items listed as follows must be set up at a supervised machine.

(a) **Setting up the directory subject to monitoring**

Execute the `hdspathlist` command at the supervised machine to set up the directory (`HSPATH` or `HDEPATH`) of Datareplicator that is to be subject to supervision. JP1/Cm2 monitors Datareplicator that corresponds to the directory specified in the `hdspathlist` command.

(b) **Creating a shell script**

To use the remote control facility to execute a Datareplicator command that requires a HiRDB environment variable, you must create a *shell script*. Specify the Datareplicator command and any HiRDB environment variables in this shell script. The following Datareplicator commands require HiRDB environment variables:

- `hdeevent` command (issues an event)
- `hdeprep` command (creates an extraction definition preprocessing file)
- `hdestart` command (starts the source Datareplicator)
- `hdsstart` command (starts the target Datareplicator)

When you specify the filename of the created shell script, the remote control facility executes the Datareplicator command in the environment specified in the shell script. There is no need to create a shell script to execute a command that does not require environment variables.

To create a shell script:

1. Copy the sample shell script into the `Datareplicator-installation-directory/bin` directory. The following are the names of the directories and files where you can find sample shell scripts:

Name of directory for sample shell scripts

`/opt/hirdbds/lib/sample`

Shell script filenames

Shell script for event issuance: hdevent_sh

Shell script for extraction definition preprocessing file: hdeprep_sh

Shell script for source Datareplicator startup: hdestart_sh

Shell script for target Datareplicator startup: hdsstart_sh

Target directory name

/opt/hirdbds/bin

2. Edit the copied file and specify the applicable command and environment variables. For details about environment variables, see *2.4 Specifying environment variables (UNIX)*.
3. Grant execution privilege to the created shell script file.

(c) Setting up the JP1/Cm2/Extensible SNMP Agent

Register the Datareplicator's MIB file (hdsMIB) at the JP1/Cm2/Extensible SNMP Agent. For details about how to set up JP1/Cm2/Extensible SNMP Agent, see the manual *JP1 Version 8 JP1/Consolidated Management 2/Extensible SNMP Agent*.

(d) Setting up the Datareplicator agent

Specify the JP1/Cm2/Extensible SNMP Agent's \$OV_BIN environment variable for the user who will execute the Datareplicator agent startup command (hdsagtstart). For automatic startup of the Datareplicator agent subject to status monitoring, register the following pathname in /etc/localrc:

Registered path name

/opt/hirdbds/lib/hdsagtstart_sh

To change settings for the Datareplicator agent settings (such as the monitoring interval), execute the hdsagtopt command.

(3) Setting up a supervised machine (for a Windows Datareplicator)

In the case of a Windows Datareplicator, the items listed as follows must be set up at a supervised machine.

(a) Setting up the SNMP service

From **Network Computer**, choose the **Services** tab to set up the SNMP service.

(b) Setting up the Datareplicator agent

For automatic startup of the Datareplicator agent subject to status monitoring, specify **HiRDB Datareplicator (Agent)** in the Windows Service dialog box as the service to be subject to automatic startup.

To change settings for the Datareplicator agent settings (such as the monitoring interval), execute the `hdsagtopt` command.

3.4.7 Operation of the supervisor machine

This section explains how to operate the supervisor machine to manipulate a Datareplicator.

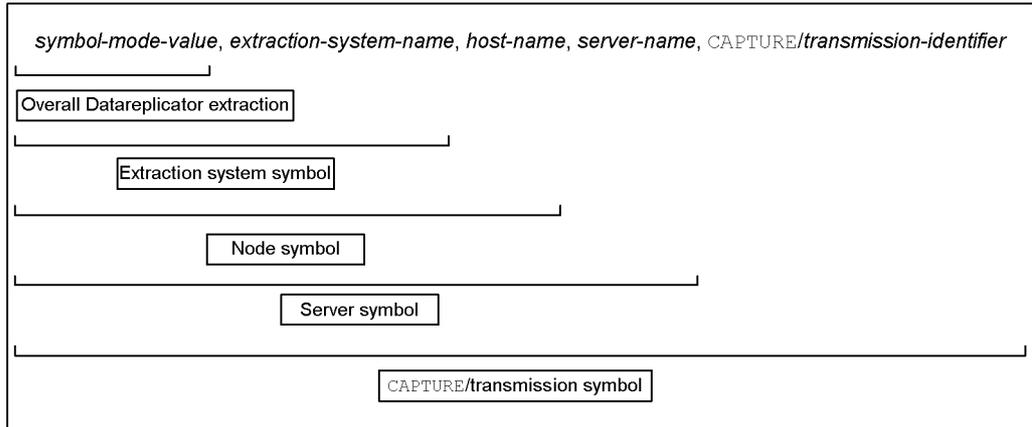
(1) Executing status monitoring

On the JP1/Cm2/Network Node Manager's screen, double-click on the machine symbol to display the Datareplicator symbol. The following are the conventions governing the symbol names displayed by the JP1/Cm2/Network Node Manager:

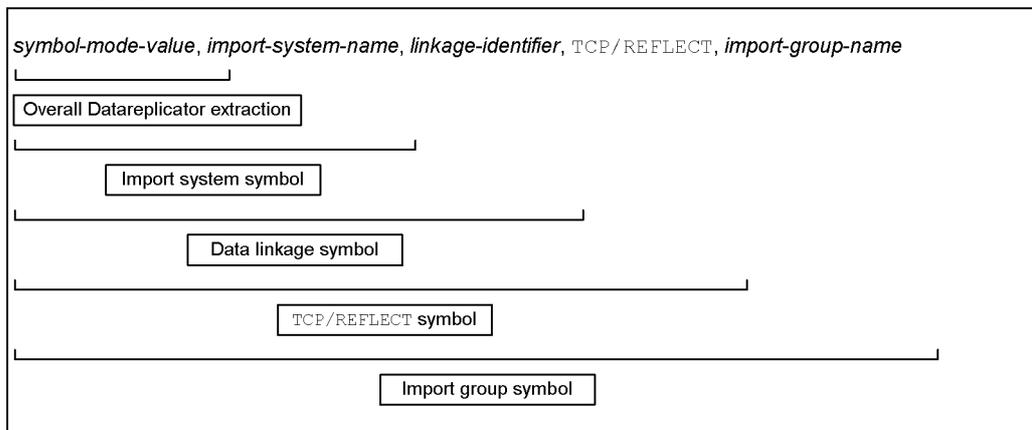
Symbol name conventions (`notree` specified)

3. Data Linkage Facilities

● For extraction processing



● For import processing



Symbol name conventions (*tree* specified)

Type of processing	Type of symbol	Name of symbol
Extraction	Overall source Datareplicator	<i>specified-symbol-mode</i>
	Extraction system identifier	hdeid: <i>XX</i>
	Host name	host: <i>XXXX</i>
	Server name	server: <i>XXXX</i>
	Extraction processing	CAPTURE
	Transmission identifier	send: <i>XXXX</i>
Import	Overall target Datareplicator	<i>specified-symbol-mode</i>
	Import system identifier	hdsid: <i>XX</i>
	Data linkage identifier	dsid: <i>XX</i>
	Communication processing	TCP
	Import processing	REFLECT
	Import group name	group: <i>XXXX</i>

Notes

The way the symbol names are displayed depends on the JP1/Cm2 version and the specification of the `hdsagtopt -d` command option. The following table shows the relationships among the JP1/Cm2 version, the specification of the `hdsagtopt -d` command option, and the display:

hdsagtopt -d command option	JP1/Cm2/Network Node Manager, JP1/Cm2/SNMP System Observer, and JP1/Cm2/Extensible SNMP Agent version	
	Earlier than 05-00	05-00 or later
tree	Invalid display	A tree structure is displayed under the machine symbol.
notree	All processes are displayed under the machine symbol.	All processes are displayed under the machine symbol.

When a tree structure is displayed, double-clicking on a symbol displays the lower-level symbols.

(2) Executing information collection

Use the JP1/Cm2/Network Node Manager's MIB browser to reference information about the current supervised machine. Specify the object ID of the desired

Datareplicator with the MIB browser to select a query.

To collect information, use the JP1/Cm2/Network Node Manager's data collection and threshold values. Specify the object ID of the desired Datareplicator. For details about Datareplicator's object ID, see 3.4.9 *MIB file*.

(3) Executing remote control

Use the JP1/Cm2/Network Node Manager's MIB browser to execute a command at a remote Datareplicator.

To execute remote control:

1. Specify in the MIB object ID the object ID for Datareplicator command issuance (`.iso.org.dod.internet.private.enterprises.hitachi.systemExMib.hirdbMibs.hirdbReplicator.hdCommand`).
2. Specify the value 0 for the MIB instance.
3. Specify in the following format the command to be executed for the SNMP value:

Format:

xn command

Rules:

x: Type of Datareplicator:

e: Source Datareplicator

s: Target Datareplicator

n: Datareplicator's index number within one machine. Specify the index number beginning with 1 for each Datareplicator, source and target. You can use the information collection facility to check index numbers.

command: Datareplicator command or filename of a created shell script.

4. Click the **Set** button.

(a) Notes

- The remote control facility does not check the validity of the specified Datareplicator command name or options. No error message is displayed if the executed command results in an error. You can use the status monitoring or information collection facility to determine whether a command has executed successfully.
- You cannot execute a command that issues a query, such as the `hdestart -i` command, because there is no way to enter a response.
- Do not specify a double quotation mark (`"`). Datareplicator will not execute a command containing a double quotation mark even if the command format is

correct. If a command's option parameter contains a space (such as the filename `Program files`), enclose the parameter in single quotation marks (').

- Information about commands executed by the remote control facility is displayed in the supervised machine's system log or event log, except for the percent sign (%). If a command executed successfully, information to that effect is output to the command log in the operation directory.

(b) Example of SNMP value

This example executes import processing for each transaction on the first target Datareplicator with data linkage identifier `a0` in the machine:

```
s1 hdsrfctl -d a0 -m trn
```

(4) Datareplicator agent operation

You can use JP1/Cm2/Network Node Manager's MIB browser to check the status of a Datareplicator agent or to manipulate it from a remote location.

(a) Checking the status

In the MIB browser's object ID, specify the Datareplicator agent's object ID (`.iso.org.dod.internet.private.enterprises.hitachi.systemExMib.hirdbMibs.hirdbReplicator.hdAgent`), and then choose a query. You can check the Datareplicator agent's status and option settings.

(b) Starting and terminating the Datareplicator agent

In the MIB browser's object ID, specify the Datareplicator agent's object ID for startup/termination

(`.iso.org.dod.internet.private.enterprises.hitachi.systemExMib.hirdbMibs.hirdbReplicator.hdAgent.hdAgentStart` or `hdAgentStop`), specify the value 0 for the instance, and then click **Set**. You can start or terminate the Datareplicator agent.

When the Datareplicator agent is started or terminated, the startup or termination message is output to the applicable machine's system log or event log.

(c) Specifying options

In the MIB browser's object ID, specify the object ID for the option you want to specify for the Datareplicator agent (such as

`.iso.org.dod.internet.private.enterprises.hitachi.systemExMib.hirdbMibs.hirdbReplicator.hdAgent.hdAgentTimer` to specify the monitoring interval). Specify the value 0 for the instance. Specify the new SNMP value (such as a time value), and then click **Set**. The Datareplicator agent's option is now specified.

A faster way of specifying options is by clicking the MIB value field immediately after displaying the Datareplicator agent's option values in (a) above.

If you change a Datareplicator agent's options, a message to that effect is output to the applicable machine's system log or event log. Note that if the percent sign (%) is specified as an option value, it is not output to the system log or event log.

3.4.8 Operation of a supervised machine

This section explains how to operate Datareplicator on a supervised machine.

(1) Starting and terminating the Datareplicator agent

To monitor the status of Datareplicator on a supervised machine, you must start the Datareplicator agent with the `hdsagtstart` command.

When a machine is shut down, the Datareplicator agent terminates automatically. If you want to suspend status monitoring while keeping Datareplicator active for some reason, you can use the `hdsagtstop` command to terminate the Datareplicator agent.

For details, see the `hdsagtstart` or `hdsagtstop` command in Chapter 7. *Command Syntax*.

(2) Displaying the status of a Datareplicator agent

You execute the `hdsagtstatus` command to display the status of a Datareplicator agent. For details about the displayed information, see the `hdsagtstatus` command in Chapter 7. *Command Syntax*.

(3) Changing Datareplicator agent settings

You use the `hdsagtopt` command to change Datareplicator agent settings. The command options determine the information to be changed, described as follows.

- This command changes the interval at which the status is monitored.
- This command changes the symbol display method.
- This command changes the first characters of the symbol names.
- The user specifies symbol names.

For details, see the `hdsagtopt` command in Chapter 7. *Command Syntax*.

3.4.9 MIB file

You must have a MIB file to use the information collection and remote control facilities during use of JP1/Cm2 for operations management (MIB stands for Management Information Base). A MIB file is a text file that stores the structures of the objects that are used with the SNMP protocol. All objects (information) are unique and identified by an object ID.

Do not edit Datareplicator's MIB file.

(1) Overview of Datareplicator's MIB file

The following table describes the rules for the MIB file used with Datareplicator. A

MIB file is created in strict accordance with these rules.

Table 3-4: Rules for MIB file used with Datareplicator

Item	Rules for Datareplicator's MIB file
Filename	The filename is hdsMIB.
Object ID	Datareplicator's object ID is .iso.org.dod.internet.private.enterprises.hitachi.systemExMib.hirdbMibs.hirdbReplicator. Its numeric representation is 1.3.6.1.4.1.116.5.24.1. This object ID is followed by an object that contains the Datareplicator information. For details about the object ID, see (2) in 3.4.9 MIB file.
Comment(DESCRIPTION)	The first line contains a comment on the object ID, and the subsequent lines contain JP1/Cm2's SNMP agent control command.
Index(INDEX)	MIB uses indexes to identify a table because it cannot hierarchize. For example, to determine a single queue file, the following four indexes are required: <ul style="list-style-type: none"> • System index • Node index • Server index • Queue file index
Access privilege(ACCESS)	This is the access privilege for the object. The object in the status information is read-only; GET/GETNEXT is granted; SET is rejected. Datareplicator's control command object is write-only; SET is granted; GET/GETNEXT is rejected. not-access is not accessible. In the case of GETNEXT, low-order information is collected.
Object type(SYNTAX)	DisplayString (ASCII) is used for characters and INTEGER or Counter64 for numeric values. For a table, SEQUENCE is used.

(2) Details about MIB

(a) hirdbReplicator group (hirdbMibs 1)

The following table shows the detailed information about MIB in the hirdbReplicator group.

Table 3-5: Detailed information about MIB in the hirdbReplicator group

ID	Object name	Description	Type	Privilege
1	hdsVersion	Datareplicator version	DisplayString	read-only
2	hdes	Number of source systems	INTEGER	read-only
3	hdeSysTbl	Extraction system table	SEQUENCE	not-access
4	hdeNodes	Number of extraction nodes	INTEGER	read-only
5	hdeNodeTbl	Extraction node table	SEQUENCE	not-access

3. Data Linkage Facilities

ID	Object name	Description	Type	Privilege
6	hdeServers	Number of source HiRDB servers	INTEGER	read-only
7	hdeServerTbl	Source HiRDB's server table	SEQUENCE	not-access
8	hdeSends	Number of send processes	INTEGER	read-only
9	hdeSendTbl	Transmission table	SEQUENCE	not-access
10	hdeQFiles	Number of extraction queue files	INTEGER	read-only
11	hdeQFileTbl	Extraction queue file table	SEQUENCE	not-access
12	hdss	Number of target systems	INTEGER	read-only
13	hdsSysTbl	Target system table	SEQUENCE	not-access
14	hdsConnects	Data linkage count	INTEGER	read-only
15	hdsConnectTbl	Data linkage table	SEQUENCE	not-access
16	hdsReflects	Number of import groups	INTEGER	read-only
17	hdsReflectTbl	Import group table	SEQUENCE	not-access
18	hdsQFiles	Number of import queue files	INTEGER	read-only
19	hdsQFileTbl	Import queue file table	SEQUENCE	not-access
20	hdCommand	Datareplicator command	DisplayString	write-only
21	hdAgent	Agent table	SEQUENCE	not-access

(b) hdeSysTbl group (hrdbReplicator 3)

The following table shows the detailed information about MIB in the hdeSysTbl group.

Table 3-6: Detailed information about MIB in the hdeSysTbl group

ID	Object name	Description	Type	Privilege
3.	hdeSysTbl	Extraction system table	SEQUENCE	not-access
3.1	hdeSysEnt	Extraction system table entry	SEQUENCE	not-access
3.1.1	hdeSysIndex	Extraction system index	INTEGER	read-only
3.1.2	hdeSysId	Extraction identifier	DisplayString	read-only
3.1.3	hdeSysPath	Name of extraction directory	DisplayString	read-only
3.1.4	hdeSysMasterStLevel	Extraction master status level	INTEGER	read-only

ID	Object name	Description	Type	Privilege
3.1.5	hdeSysMasterStatus	Details about extraction master status	DisplayString	read-only
3.1.6	hdeSysNodes	Number of nodes for the corresponding extraction system	INTEGER	read-only

(c) hdeNodeTbl group (hirdbReplicator 5)

The following table shows the detailed information about MIB in the hdeNodeTbl group.

Table 3-7: Detailed information about MIB in the hdeNodeTbl group

ID	Object name	Description	Type	Privilege
5.	hdeNodeTbl	Extraction node table	SEQUENCE	not-access
5.1	hdeNodeEnt	Extraction node table entry	SEQUENCE	not-access
5.1.1	hdeNodeSysIndex	Extraction system index	INTEGER	read-only
5.1.2	hdeNodeIndex	Extraction node index	INTEGER	read-only
5.1.3	hdeNodeId	Host name	DisplayString	read-only
5.1.4	hdeNodeMastStLevel	Node master status level	INTEGER	read-only
5.1.5	hdeNodeMastStatus	Details about node master status	DisplayString	read-only
5.1.6	hdeNodeServers	Number of target HiRDB servers for the corresponding node	INTEGER	read-only

(d) hdeServerTbl group (hirdbReplicator 7)

The following table shows the detailed information about MIB in the hdeServerTbl group.

Table 3-8: Detailed information about MIB in the hdeServerTbl group

ID	Object name	Description	Type	Privilege
7.	hdeServerTbl	Source HiRDB server table	SEQUENCE	not-access
7.1	hdeServerEnt	Source HiRDB server table entry	SEQUENCE	not-access
7.1.1	hdeServerSysIndex	Extraction system index	INTEGER	read-only
7.1.2	hdeServerNodeIndex	Node index	INTEGER	read-only

ID	Object name	Description	Type	Privilege
7.1.3	hdeServerIndex	Source HiRDB server index	INTEGER	read-only
7.1.4	hdeServerId	Source HiRDB server name	DisplayString	read-only
7.1.5	hdeServerCaputureStLevel	Extraction status level	INTEGER	read-only
7.1.6	hdeServerCaputureStatus	Details about extraction status	DisplayString	read-only
7.1.7	hdeServerSends	Number of send processes for the corresponding server	INTEGER	read-only
7.1.8	hdeServerQFiles	Number of queue files for the corresponding server	INTEGER	read-only
7.1.9	hdeServerWriteQFNo	Write queue file number for the corresponding server	INTEGER	read-only
7.1.10	hdeServerWriteQFOffset	Write queue file offset location for the corresponding server	Counter64	read-only

(e) hdeSendTbl group (hirdbReplicator 9)

The following table shows the detailed information about MIB in the hdeSendTbl group.

Table 3-9: Detailed information about MIB in the hdeSendTbl group

ID	Object name	Description	Type	Privilege
9.	hdeSendTbl	Transmission table	SEQUENCE	not-access
9.1	hdeSendEnt	Transmission table entry	SEQUENCE	not-access
9.1.1	hdeSendSysIndex	Extraction system index	INTEGER	read-only
9.1.2	hdeSendNodeIndex	Extraction node index	INTEGER	read-only
9.1.3	hdeSendServerIndex	Source HiRDB server index	INTEGER	read-only
9.1.4	hdeSendIndex	Transmission index	INTEGER	read-only
9.1.5	hdeSendId	Transmission name	DisplayString	read-only
9.1.6	hdeSendProcStLevel	Transmission status level	INTEGER	read-only

ID	Object name	Description	Type	Privilege
9.1.7	hdeSendProcStatus	Details about transmission status	DisplayString	read-only
9.1.8	hdeSendReadQNo	Read queue file number	INTEGER	read-only
9.1.9	hdeSendReadQOffset	Read queue file offset location	Counter64	read-only
9.1.10	hdeSendIns	Insert count	Counter64	read-only
9.1.11	hdeSendUpd	Update count	Counter64	read-only
9.1.12	hdeSendDel	Delete count	Counter64	read-only
9.1.13	hdeSendPurge	Purge count	Counter64	read-only
9.1.14	hdeSendEvent	Number of events	Counter64	read-only
9.1.15	hdeSendTran	Number of transmission transactions	Counter64	read-only
9.1.16	hdeSendUndetermTran	Number of unresolved transactions	Counter64	read-only

(f) hdeQFTbl group (hirdbReplicator 11)

The following table shows the detailed information about MIB in the hdeQFTbl group.

Table 3-10: Detailed information about MIB in the hdeQFTbl group

ID	Object name	Description	Type	Privilege
11.	hdeQFTbl	Queue file table	SEQUENCE	not-access
11.1	hdeQFEnt	Queue file table entry	SEQUENCE	not-access
11.1.1	hdeQFSysIndex	Extraction system index	INTEGER	read-only
11.1.2	hdeQFNodeIndex	Extraction node index	INTEGER	read-only
11.1.3	hdeQFServerIndex	Source HiRDB server index	INTEGER	read-only
11.1.4	hdeQFIndex	Extraction queue file index	INTEGER	read-only
11.1.5	hdeQFId	Extraction queue file ID	DisplayString	read-only
11.1.6	hdeQFUseSize	Current size of extraction queue file used	Counter64	read-only
11.1.7	hdeQFSize	Size of extraction queue file	Counter64	read-only

(g) hdsSysTbl group (hirdbReplicator 13)

The following table shows the detailed information about MIB in the hdsSysTbl group.

Table 3-11: Detailed information about MIB in the hdsSysTbl group

ID	Object name	Description	Type	Privilege
13.	hdsSysTbl	Import system table	SEQUENCE	not-access
13.1	hdsSysEnt	Import system table entry	SEQUENCE	not-access
13.1.1	hdsSysIndex	Import system index	INTEGER	read-only
13.1.2	hdsSysId	Import identifier	DisplayString	read-only
13.1.3	hdsSysPath	Name of import directory	DisplayString	read-only
13.1.4	hdsSysMasterStLevel	Import master status level	INTEGER	read-only
13.1.5	hdsSysMasterStatus	Details about import master status	DisplayString	read-only
13.1.6	hdsSysConnects	Data linkage count for the corresponding import system	Counter64	read-only

(h) hdsConnectTbl group (hirdbReplicator 15)

The following table shows the detailed information about MIB in the hdsConnectTbl group.

Table 3-12: Detailed information about MIB in the hdsConnectTbl group

ID	Object name	Description	Type	Privilege
15.	hdsConnectTbl	Data linkage table	SEQUENCE	not-access
15.1	hdsConnectEnt	Data linkage table entry	SEQUENCE	not-access
15.1.1	hdsConnectSysIndex	Import system index	INTEGER	read-only
15.1.2	hdsConnectIndex	Data linkage index	INTEGER	read-only
15.1.3	hdsConnectId	Data linkage identifier	DisplayString	read-only
15.1.4	hdsConnectTCPMastStLevel	Communication status level	INTEGER	read-only
15.1.5	hdsConnectTCPMastStatus	Details about communication status	DisplayString	read-only
15.1.6	hdsConnectDefServeStLevel	Import definition server status level	INTEGER	read-only

ID	Object name	Description	Type	Privilege
15.1.7	hdsConnectDefServeStatus	Details about import definition server status	DisplayString	read-only
15.1.8	hdsConnectReflectGroups	Number of import groups for the corresponding data linkage	INTEGER	read-only
15.1.9	hdsConnectQFiles	Number of queue files for the corresponding data linkage	INTEGER	read-only
15.1.10	hdsConnectWriteQFNo	Write queue file number for the corresponding data linkage	INTEGER	read-only
15.1.11	hdsConnectWriteQFOffset	Write queue file offset location for the corresponding data linkage	INTEGER	read-only

(i) hdsReflectTbl group (hirdbReplicator 17)

The following table shows the detailed information about MIB in the hdsReflectTbl group.

Table 3-13: Detailed information about MIB in the hdsReflectTbl group

ID	Object name	Description	Type	Privilege
17.	hdsReflectTbl	Import group table	SEQUENCE	not-access
17.1	hdsReflectEnt	Import group table entry	SEQUENCE	not-access
17.1.1	hdsReflectSysIndex	Import system index	INTEGER	read-only
17.1.2	hdsReflectConnectIndex	Data linkage index	INTEGER	read-only
17.1.3	hdsReflectIndex	Import group index	INTEGER	read-only
17.1.4	hdsReflectType	Import group type	INTEGER	read-only
17.1.5	hdsReflectId	Import group name	DisplayString	read-only
17.1.6	hdsReflectGroupStLevel	Import group status level	INTEGER	read-only
17.1.7	hdsReflectGroupStatus	Details about import group status	DisplayString	read-only
17.1.8	hdsReflectGrpReadQNo	Read queue file number	INTEGER	read-only

ID	Object name	Description	Type	Privilege
17.1.9	hdsReflectGrpReadQOffset	Read queue file offset location	Counter64	read-only
17.1.10	hdsReflectGrpIns	Insert count	Counter64	read-only
17.1.11	hdsReflectGrpUpd	Update count	Counter64	read-only
17.1.12	hdsReflectGrpDel	Delete count	Counter64	read-only
17.1.13	hdsReflectGrpPurge	Purge count	Counter64	read-only
17.1.14	hdsReflectGrpCommit	Commit count	Counter64	read-only
17.1.15	hdsReflectGrpTimestamp	Time stamp	Counter64	read-only
17.1.16	hdsReflectGrpTran	Number of read transactions	Counter64	read-only

(j) hdsQFTbl group (hirdbReplicator 19)

The following table shows the detailed information about MIB in the hdsQFTbl group.

Table 3-14: Detailed information about MIB in the hdsQFTbl group

ID	Object name	Description	Type	Privilege
19.	hdsQFTbl	Import queue file table	SEQUENCE	not-access
19.1	hdsQFEnt	Import queue file table entry	SEQUENCE	not-access
19.1.1	hdsQFSysIndex	Import system index	INTEGER	read-only
19.1.2	hdsQFConnectIndex	Data linkage index	INTEGER	read-only
19.1.3	hdsQFIndx	Import queue file index	INTEGER	read-only
19.1.4	hdsQFId	Name of import queue file	DisplayString	read-only
19.1.5	hdsQFUseSize	Current size of import queue file used	Counter64	read-only
19.1.6	hdsQFSize	Size of import queue file	Counter64	read-only

(k) hdAgent group (hirdbReplicator 21)

The following table shows the detailed information about MIB in the hdAgent group.

Table 3-15: Detailed information about MIB in the hdAgent group

ID	Object name	Description	Type	Privilege
21.	hdAgent	Agent table	N/A	not-access
21.1	hdAgentTimer	Monitoring interval	INTEGER	read-write
21.2	hdAgentSymbolDisplay	Display method	DisplayString	read-write
21.3	hdAgentSymbolMode	Symbol mode	DisplayString	read-write
21.4	hdAgentSourceSymbol	Extraction user symbol	DisplayString	read-write
21.5	hdAgentTargetSymbol	Import user symbol	DisplayString	read-write
21.6	hdAgentStatus	Agent status	DisplayString	read-only
21.7	hdAgentStart	Agent startup	DisplayString	write-only
21.8	hdAgentStop	Agent termination	DisplayString	write-only

3.5 Datareplicator file system areas

A Datareplicator file system area provides in a character special file an efficient method for storing files used by Datareplicator.

Note

Datareplicator file system areas are available only when the UNIX Datareplicator is used. You cannot use Datareplicator file system areas for the Windows Datareplicator.

3.5.1 Purpose of a Datareplicator file system area

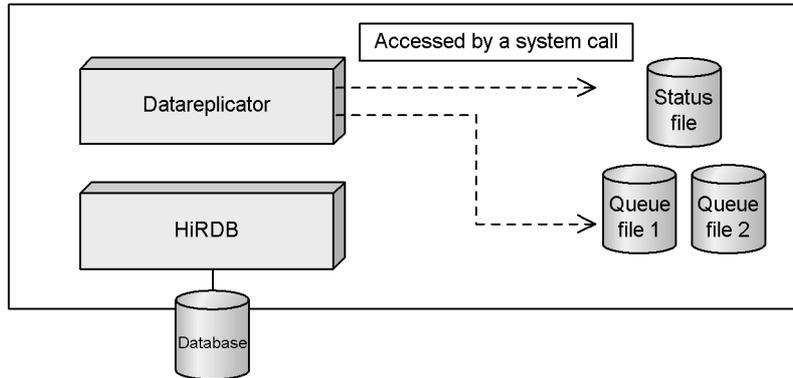
The minimum size of a UNIX character special file is 1MB. If you allocate a source Datareplicator file to a UNIX character special file, the storage efficiency would be poor because there would be more space than needed. The Datareplicator file system area makes it possible to manage file areas efficiently, avoiding poor storage efficiency.

By defining Datareplicator file system areas, you can place multiple files used by Datareplicator (including system files) in a single character special file. The following figure shows the relationship between a Datareplicator file system area and Datareplicator files.

Figure 3-31: Relationship between a Datareplicator file system area and Datareplicator files

- Normal Datareplicator system files

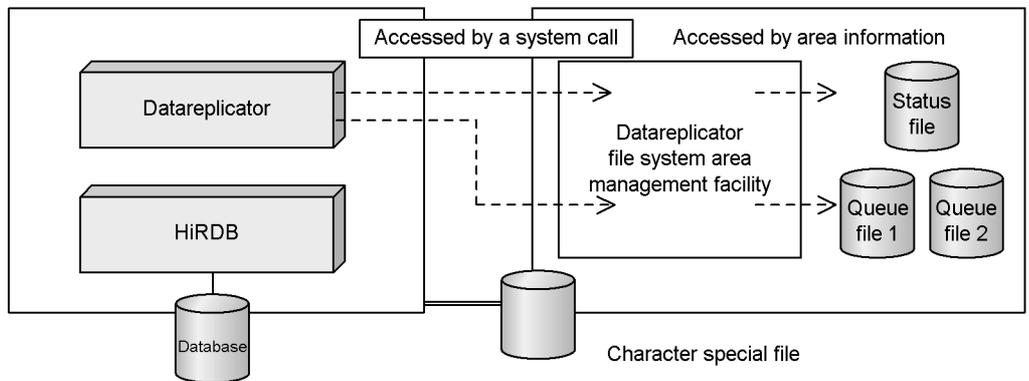
Datareplicator system



- When a Datareplicator file system area is used

Datareplicator system

Datareplicator file system area



(1) Files that can be stored in Datareplicator file system areas

The following table lists the system files that can be stored in Datareplicator file system areas.

Table 3-16: Files that can be stored in Datareplicator file system areas

Type of Datareplicator	Files that can be stored
Source Datareplicator	Extraction server status file
	Extraction information queue files
	Data linkage file
Target Datareplicator	Import status files
	Import information queue files

3.5.2 Creating a Datareplicator file system area

This section explains the procedure for creating a Datareplicator file system area.

(1) Creating a new Datareplicator file system area

To create a new Datareplicator file system area:

1. Prepare a character special file that that can be allocated as a Datareplicator file system area.
2. Initialize the character special file as a Datareplicator file system area.
Use the `hdsfmkfs` command for this purpose.
3. Specify the `devicexx` operand in Datareplicator's extraction environment definition or import environment definition.
4. Allocate system files to the Datareplicator file system area.

(2) Changing an existing Datareplicator file system area

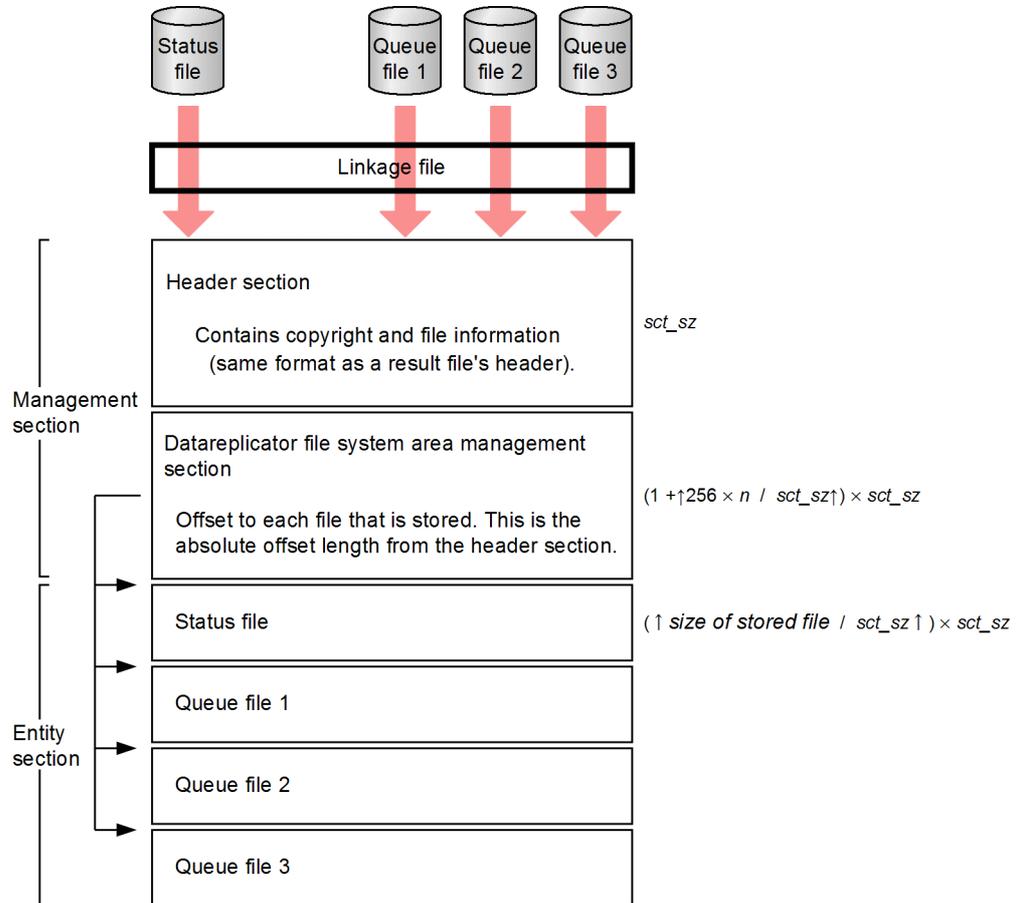
- When Datareplicator's directory remains unchanged
Initialize Datareplicator (execute the `hdsstart -i` command).
- When Datareplicator's directory is different than before
After initializing the Datareplicator file system area with the `hdsfmkfs` command, execute initial startup of Datareplicator (execute the `hdsstart -i` command).

The following figure shows a Datareplicator file system area's status changes.

area, an entity section that stores allocated files, and the files that are linked to the Datareplicator file system area. Processes and commands use these linked filenames to access the system files allocated to the Datareplicator file system area.

The following figure shows the structure of a Datareplicator file system area.

Figure 3-33: Structure of a Datareplicator file system area



Legend:

sct_sz: Sector size of the Datareplicator file system area

(1) Rules for allocating a Datareplicator file system area

The following rules apply to allocating system files to the entity section of a Datareplicator file system area:

- The number of allocated system files cannot exceed the number of files specified

in the `-l` option of the `hdsfmkfs` command.

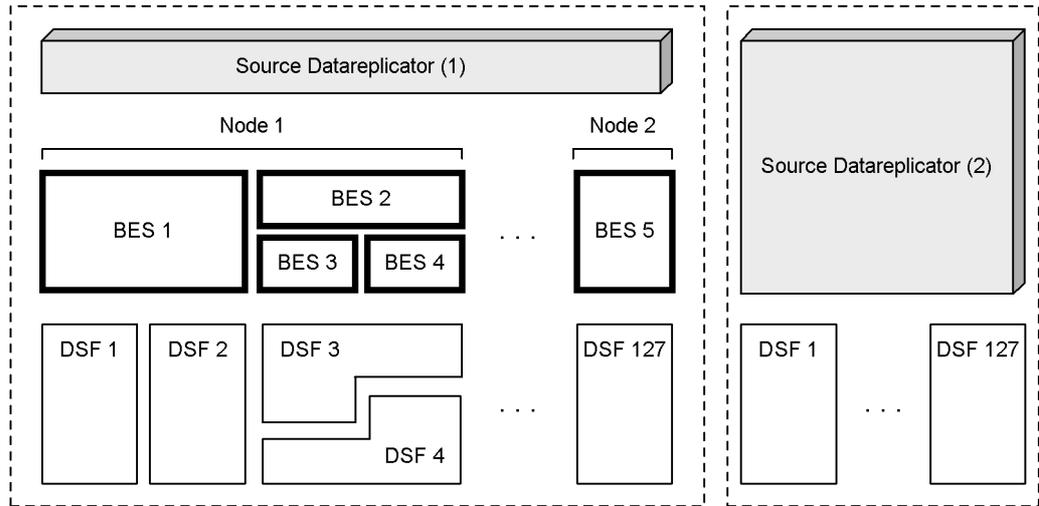
- The total size of all files allocated to a single Datareplicator file system area must be within the following range:

$\begin{aligned} & \text{Size of a Datareplicator file system area} \geq \text{sct_sz} && \text{(Header section)} \\ & + (1 + \lceil 256 * n / \text{sct_sz} \rceil) * \text{sct_sz} && \text{(Management section)} \\ & + \sum n ((\lceil \text{Size of stored file} / \text{sct_sz} \rceil) * \text{sct_sz}) && \text{(Cumulative allocation file size)} \end{aligned}$
--

`sct_sz` : Sector length of the Datareplicator file system area
`n` : Number of files stored

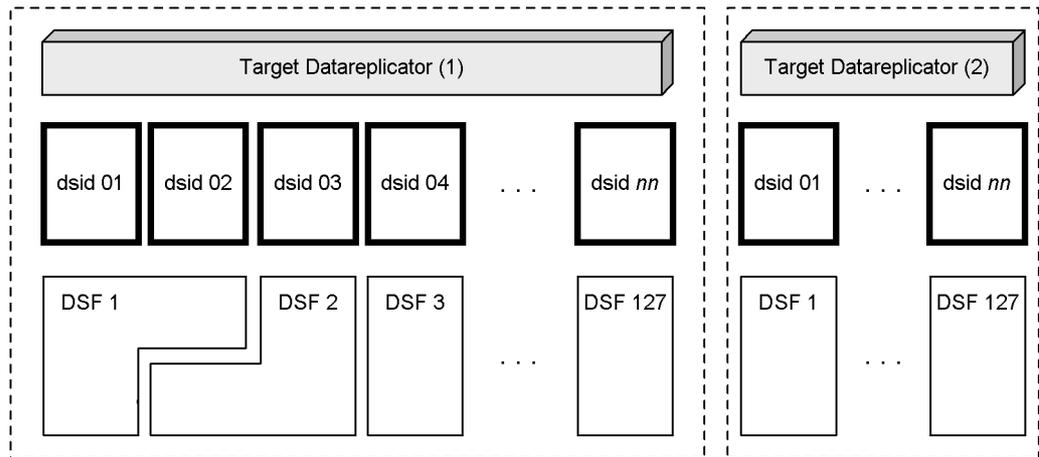
- A single system file cannot be divided and allocated to multiple Datareplicator file system areas.
- Datareplicator files under different directories cannot be allocated to the same Datareplicator file system area.
- Multiple system files in one Datareplicator used for one machine must be allocated to the same Datareplicator file system area. Figures 3-34 and 3-35 show examples of allocation.

Figure 3-34: Example of storage in Datareplicator file system areas in the source system



BES: Back-end server
 DSF: Datareplicator file system area

Figure 3-35: Example of storage in Datareplicator file system areas in the target system



DSF: Datareplicator file system area

- If multiple data linkage identifiers (`dsid`) are defined at the target Datareplicator, you need to allocate a Datareplicator file system area for each data linkage identifier to execute partial initial startup. If the same Datareplicator file system area is shared among multiple data linkage identifiers, partial initial startup will result in an error.

3.5.4 Notes on using Datareplicator file system areas

Note the following about using Datareplicator file system areas:

- If multiple data linkage identifiers (`dsid`) are defined at the target Datareplicator, create on a separate disk a Datareplicator file system area for each data linkage identifier. When you allocate files in this manner, you distribute disk I/O operations.
- If there are multiple back-end servers in the same node at the source Datareplicator, create on a separate disk a Datareplicator file system area for each back-end server. When you do this, you distribute disk I/O operations.

3.6 Delay monitoring facility

You use the *delay monitoring facility* to monitor the update time difference at the source and target Datareplicators.

3.6.1 Overview of the delay monitoring facility

The delay monitoring facility monitors the delay period for update data. The *delay period* is the difference between the time at which the source DB's transaction commits (and the update information is stored in the system log) and the time at which data linkage processing is completed. Figure 3-36 shows the concept of the delay monitoring facility. Table 3-17 lists and describes the items that are subject to delay monitoring.

Figure 3-36: Concept of the delay monitoring facility

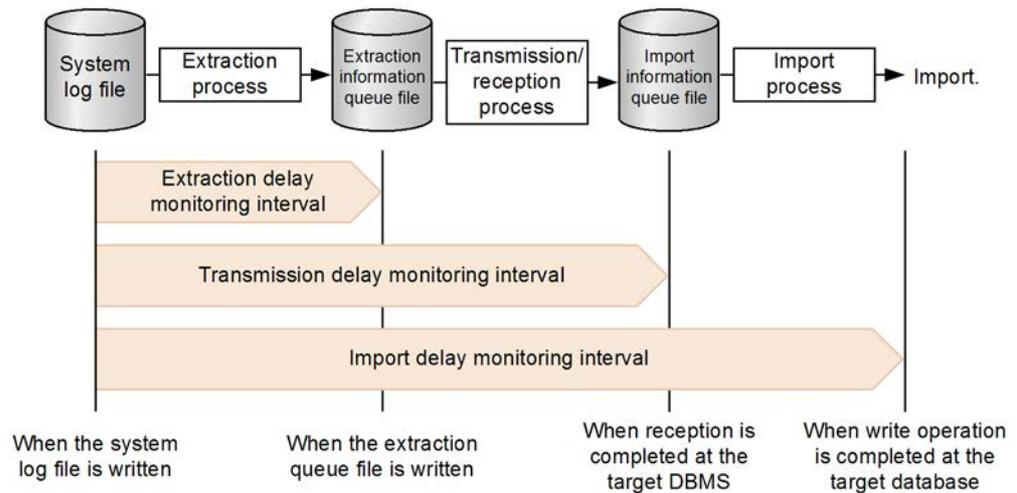


Table 3-17: Items subject to delay monitoring

Item that is monitored	Description	Monitored process
Extraction delay period	<p>This is the difference between the time at which update information is stored in the system log file and the time at which the update information is written into the extraction information queue file.</p> <p>This item is monitored when update information is written into the extraction information queue file. If there are multiple update information items in the extraction information queue I/O buffer for extraction, the extraction delay period is the difference between the time at which update information is stored in the system log file and the time at which the first committed update information is written.</p>	Extraction process
Transmission delay period	<p>This is the difference between the time at which update information is stored in the system log file (and the update information is sent) and the time at which its reception is completed at the target DBMS.</p> <p>This item is monitored for each transmission interval of the transmission process. If multiple update information items are sent in the batch mode, this is the difference between the time at which update information is stored in the system log file and the time at which reception of the first update information sent is completed.</p>	Transmission process
Import delay period	<p>This is the difference between the time at which update information is stored in the system log file and the time at which the update information is imported into the target database.</p> <p>This item is monitored for each commit interval at the target database.</p>	Import process

Monitoring of the extraction delay period is not supported in Datareplicator Extension. For details about Datareplicator Extension's delay monitoring facility, see the manual *HiRDB Datareplicator Extension Version 8*.

3.6.2 Using the delay monitoring facility

(1) Relationship between operand specification and message output

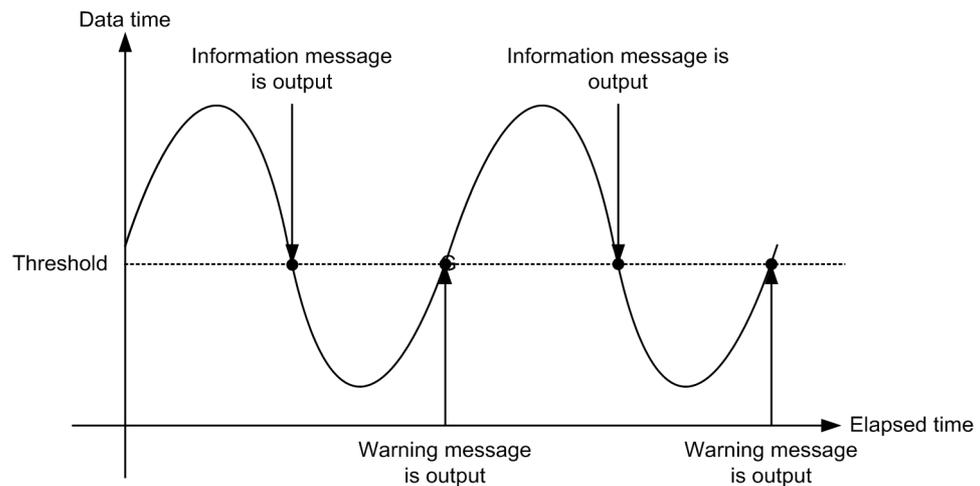
If you will be using the delay monitoring facility, specify the operands listed below.

For details about the operands, see 5.3 *Extraction environment definition*, 5.4 *Transmission environment definition*, and 5.9 *Import environment definition*.

- `extract_delay_limit_time` operand
- `send_delay_limit_time` operand
- `reflect_delay_limit_time` operand

When the delay monitoring facility is used, a warning message is output when a threshold specified in any of these operands is exceeded, as well as when the value returns to within the threshold. Only one warning message is output when a threshold is exceeded, and only one message is output when a value is within the threshold. The following figure shows the message output timing.

Figure 3-37: Message output timing



(2) Delay monitoring by the source system's `watchintvl` operand

In the source DB, the extraction master process outputs the delay status at each node at the delay monitoring interval specified in the `watchintvl` operand. If a threshold is exceeded, the delay monitoring facility outputs the `KFRB00700-W (00066)` message. When the value returns to within the threshold after having been exceeded, the facility outputs the `KFRB00700-W (00067)` message. No additional message is output as long as the status remains above the threshold or remains within the threshold.

For details about the operand definition, see 5.3 *Extraction environment definition*.

Depending on the delay status and `watchintvl` timing, the same message might be output more than once in succession, or the `KFRB00700-W (00067)` message might be output before the `KFRB00700-W (00066)` message is output.

(3) Warning message output destination

The delay monitoring facility's warning messages are output to the error information file and the syslog file. The table below lists the warning message output destinations and message numbers for the monitored items. For details about the messages that are output, see 10.2 *Details about messages*

Table 3-18: Warning message output destination and message number for each monitored item

Monitored item	Output location	Output destination	Message number
<ul style="list-style-type: none"> • Extraction delay period • Transmission delay period 	Server machine where delay occurred	<ul style="list-style-type: none"> • syslog file • Error information file 	KFRB00066-W
	Server machine where the source HiRDB's manager is running	<ul style="list-style-type: none"> • syslog file • master error information file 	KFRB00700-W
Import delay period	Server machine where import processing takes place	<ul style="list-style-type: none"> • syslog file • Error information file 	KFRB00066-W

3.6.3 Notes

The notes in this subsection apply to using the delay monitoring facility.

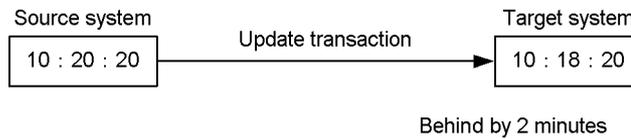
- You use a status information collection command (`hdstate` or `hdsstate`) to view the delay period. This means that Datareplicator must be running in order to view the delay period.
- The delay monitoring facility is applicable only when extraction, transmission, and import processes are running. The facility does not monitor a delay period for a process that is stopped.
- The delay monitoring facility supports only Japan Standard Time (JST-9).
- If data is imported from a PDMII SAM file, the facility monitors the import delay period beginning at the time of execution of the `hdssamqin` command.
- Do not use the delay monitoring facility in an environment in which some transactions require a large differential between the update time of the last updating SQL statement and the time of commit processing. Even if the extraction

delay period has not resulted in an exceeded delay (status in which the operand's threshold is exceeded), the transmission delay period or import delay period might result in an exceeded delay.

- If there is a mismatch between the times at the source and target systems, the facility cannot provide accurate delay period monitoring.

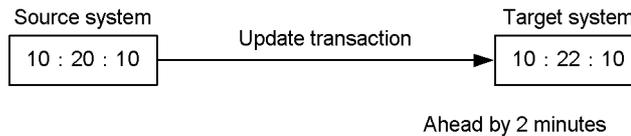
To illustrate this note about the delay monitoring facility, Figures 3-38 and 3-39 show examples in which 1 minute is specified as the threshold for the import delay period.

Figure 3-38: When the target system's time is behind



A warning message is output whenever it takes 1 minute or more for import processing to be completed after update information is stored in the system log file. In the case of the environment shown in Figure 3-38, the target system's system time is 2 minutes behind the system time at the source system. Only if it takes 3 minutes or more until import processing is completed will a warning message be output. In this case, you can adjust the target system's time to match the source system's time, in order to obtain the expected processing based on the set threshold. To change the time, you must terminate the related products that are running. If you are backdating the target system's time, you must start the related products after they have caught up with the time to be set.

Figure 3-39: When the target system's time is ahead



As in the environment shown in Figure 3-38, a warning message is output whenever it takes 1 minute or more until import processing is completed after update information is stored in the system log file. In this example, the target system's system time is 2 minutes ahead of the system time at the source system, so the warning message is output even before import processing is completed. In this case, you can prevent output of the warning message by adjusting the target system's time to match the time at the source system.

Supplement:

To avoid cases such as those shown in these examples, we recommend that when

there is a time difference between the source and target systems, you make the necessary adjustments to the system times in advance. If you will not be adjusting the system times in cases such as these, specify a threshold that takes into account the time difference between the source and target systems.

3.7 Import transaction synchronization facility

This section describes the import transaction synchronization facility.

We recommend that you read *3.7.8 Notes* before you use this facility.

3.7.1 Overview of the import transaction synchronization facility

The two types of transactions that can occur in HiRDB are transactions that occur separately at each back-end server (forming transaction branches) and transactions that combine them (called global transactions). Normally, Datareplicator links to one transaction branch at a time; therefore, immediately after a transaction branch is imported, data integrity cannot be guaranteed for a global transaction.

When the import transaction synchronization facility is used, data integrity for each global transaction that has occurred at the source HiRDB can be guaranteed when transaction branches are imported.

3.7.2 Preparations for the import transaction synchronization facility

(1) Prerequisites

You can use the import transaction synchronization facility only when the source and target systems are both Datareplicator.

To use this facility, all of the following conditions must be satisfied:

- The source HiRDB's version is 07-03 or later and it is a HiRDB/Parallel Server.
- The source Datareplicator's version is 07-04 or later.
- The target Datareplicator's version is 07-04 or later.

If you are using any of the following OSs, the target HiRDB's version must be 07-03 or later:

- HP-UX (IPF)
- Linux (IPF)
- Windows

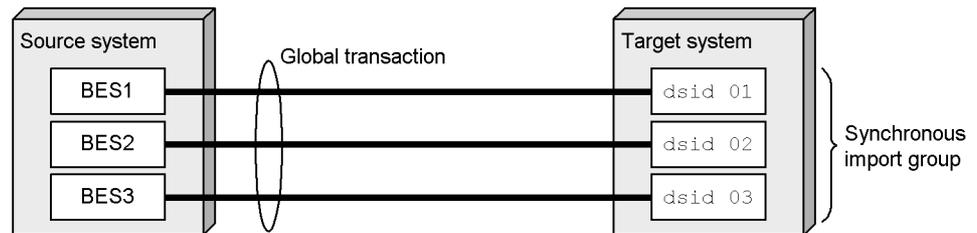
If the system uses HiRDB Datareplicator Extension, data integrity cannot be guaranteed for a global transaction because HiRDB Datareplicator Extension does not support the XA interface.

(2) System configuration**(a) Correspondence between the source system's back-end servers and the target system's data linkage identifiers**

To use this facility, you must provide a data linkage identifier for each back-end server of the source HiRDB. In this case, the data linkage identifiers must be provided as a *synchronous import group*. You must also place an event control table (`hde_dtbl`) at the source back-end servers.

The following figure shows an example of a configuration when the import transaction synchronization facility is used.

Figure 3-40: Example of a configuration when the import transaction synchronization facility is used



Legend:

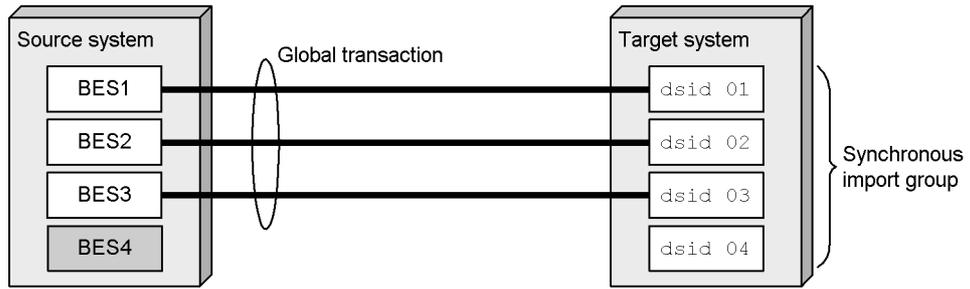
BES: Back-end server

dsid: Data linkage identifier

A global transaction that is extracted from a single source HiRDB must be received by a single target system. Therefore, the maximum number of back-end servers where a table subject to data linkage is placed is equal to the maximum number of global transactions that can be received by a single target Datareplicator (if the target system is UNIX, the maximum is 128; if it is Windows, the maximum is 63).

A back-end server that is not subject to extraction is not a target of a global transaction. As shown in the figure below, if a synchronous import group includes a data linkage identifier that corresponds to a back-end server that is not subject to extraction, import processing does not progress because the system waits for synchronization.

Figure 3-41: Example of configuration where the system waits for synchronization



Legend:

BES : Back-end server

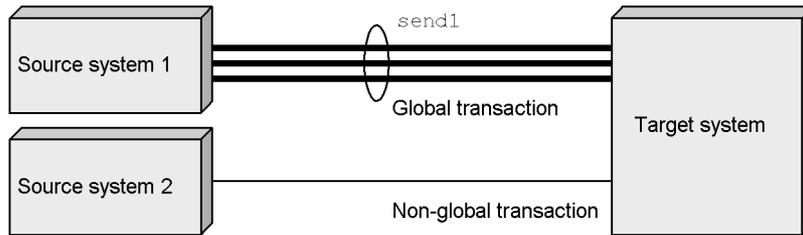
dsid : Data linkage identifier

: Back-end server that is not subject to extraction

(b) Configuration of source and target systems

When this facility is used, a single target system can receive only one global transaction. A single target system can receive both global and non-global transactions. The following figure shows an example of a configuration.

Figure 3-42: Example of a configuration of source and target systems (1)

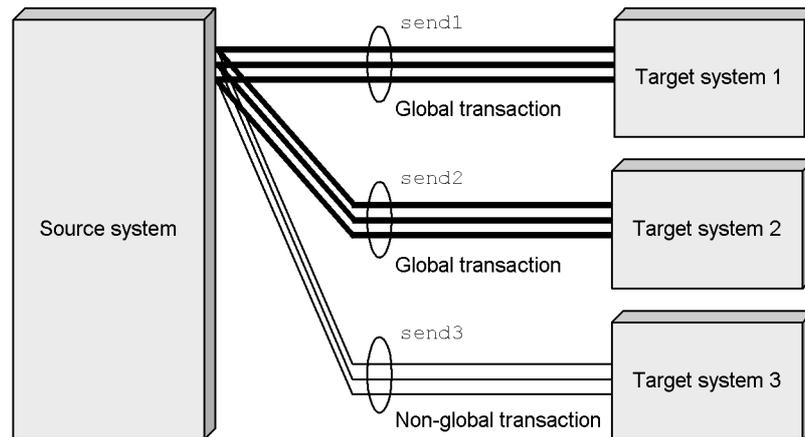


Description

This example sends to a single target system a global transaction from source system 1 and a non-global transaction from source system 2.

A single global transaction can also be sent to multiple target identifiers. You can specify whether a global transaction is to be sent for each target identifier. The following figure shows an example of a configuration.

Figure 3-43: Example of a configuration of source and target systems (2)

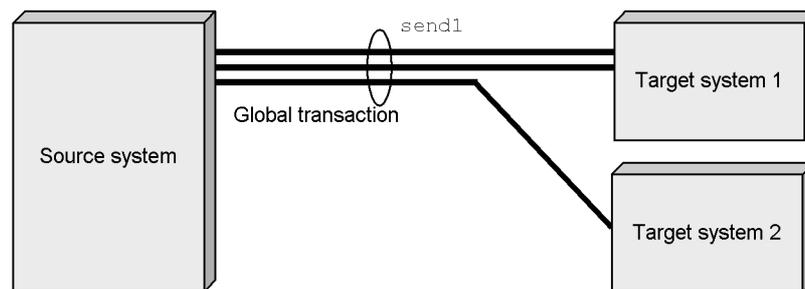


Description

This example sends global and non-global transactions from a single source system to multiple target systems.

As shown in the figure below, a set of target identifiers cannot be distributed to multiple target systems. With this configuration, synchronization is not guaranteed.

Figure 3-44: Example of a configuration that cannot be synchronized



(3) Operands to be specified

The following table shows the operands that must be specified in order to use the import transaction synchronization facility.

Table 3-19: Operands to be specified in order to use the import transaction synchronization facility

PP	Definition file	Operand	Description
HiRDB	System common definition	<code>pd_rpl_reflect_mode = uap</code> ^{#1}	Outputs transaction branch information (needed for Datareplicator to achieve this facility) to the system log file. For details, see the manual <i>HiRDB Version 9 System Definition</i> .
Datareplicator	Import system definition	<code>syncgroup001</code> ^{#2}	Constitutes a synchronous import group.
	Transmission environment definition	<code>reflect_mode = uap</code> ^{#3}	Sends transaction branch information to the target Datareplicator.
		<code>eventsync</code>	Specifies the number of the synchronous event (that triggers issuance of COMMIT by the synchronous import group).

#1

If you change the value from `uap` to `server` or vice versa during data linkage, the `KFRB03312-E` message might be output during import processing, resulting in termination of import processing. In such a case, transmission of the transaction branch information stops. To avoid this, you must release the synchronous import group, and then restart.

#2

If transaction branch information is to be sent but the `syncgroup001` operand is not defined, the `KFRB02066-W` message is output and import processing is executed asynchronously. If transaction branch information is not to be sent but the `syncgroup001` operand is defined, a port check error (default code = 26) results.

#3

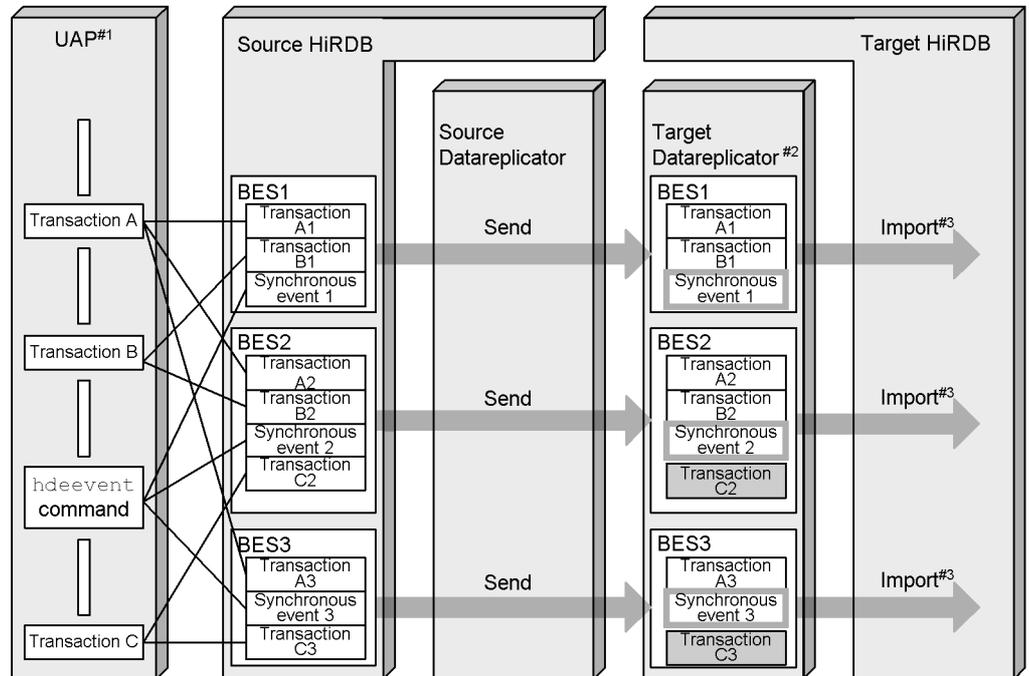
If HiRDB's `pd_rpl_reflect_mode` operand value is `server`, an error occurs during execution of the `hdestart` command and the `KFRB00726-E` message is output even when `uap` has been specified in the `reflect_mode` operand.

If you specify `uap` in the `reflect_mode` operand, specify `false` in the `overwrite` operand in the transmission environment definition. If the specified value is not `false`, the `KFRB00846-E` message is output, resulting in an error.

3.7.3 Flow of processing by the import transaction synchronization facility

The import transaction synchronization facility is achieved by using the event issuing facility. The following figure shows the flow of processing.

Figure 3-45: Flow of processing by the import transaction synchronization facility



Legend:

Transaction A: Transaction that performs update processing on BES1 through BES3
 Transaction B: Transaction that performs update processing on BES1 and BES2
 Transaction C: Transaction that performs update processing on BES2 and BES3
 BES: Back-end server

□ : Timing of COMMIT issuance

■ : Transaction that is not subject to import processing

#1

The UAP executes in the following sequence:

1. Transaction A
2. Transaction B

3. `hdeevent` command

4. Transaction C

The `hdeevent` command specifies as the event code the number of the synchronous event that has been specified in the `eventsync` operand in the transmission environment definition.

#2

Importing of the update information for `BES1`, `BES2`, and `BES3` is executed by different import processes. Therefore, importing of transactions A and B is executed asynchronously. When importing of transactions A and B is executed, `COMMIT` has not been issued yet.

#3

The import process waits for another import process when it has imported the synchronous event. When all import processes at `BES1` through `BES3` have imported all synchronous events and data integrity has been achieved at all back-end servers (after all global transactions are completed), all import processes issue `COMMIT` at the same time. Therefore, `COMMIT` for transaction C is not issued until the next synchronous event is executed.

Note:

When this facility is used, only a synchronous event triggers committing of a transaction that is generated from the target Datareplicator (*import transaction*). Therefore, unless a synchronous event is issued periodically, the import transaction becomes large and a shortage of HiRDB resources might occur. You must take into account the resources of the target HiRDB and execute a synchronous event periodically so that an import transaction does not result in an error.

A sample shell script for issuing a synchronous event periodically is shown below. This sample issues a synchronous event at an interval of five seconds.

```

#!/bin/sh
if [ -z "$1" ]
then
    echo "Please specify syncevent number."
    exit 0
fi
if [ $1 -le 0 -o $1 -ge 129 ]
then
    echo "Please specify numerical value from 1 to 128. "
    exit 0
fi
if [ -z $PDUSER ]
then
    echo "Please set PDUSER."
    exit 0
fi
echo "Start event of synchronization."
execf=OK
while [ -n $execf ]
do
    /opt/hirdbds/bin/hdeevent -n $1
    if [ $? = 0 ]
    then
        echo "event $1 O.K"
    else
        echo "Execution of hdeevent failed."
        exit 0
    fi
    sleep 5
done
echo "Stop event of synchronization."
exit 1

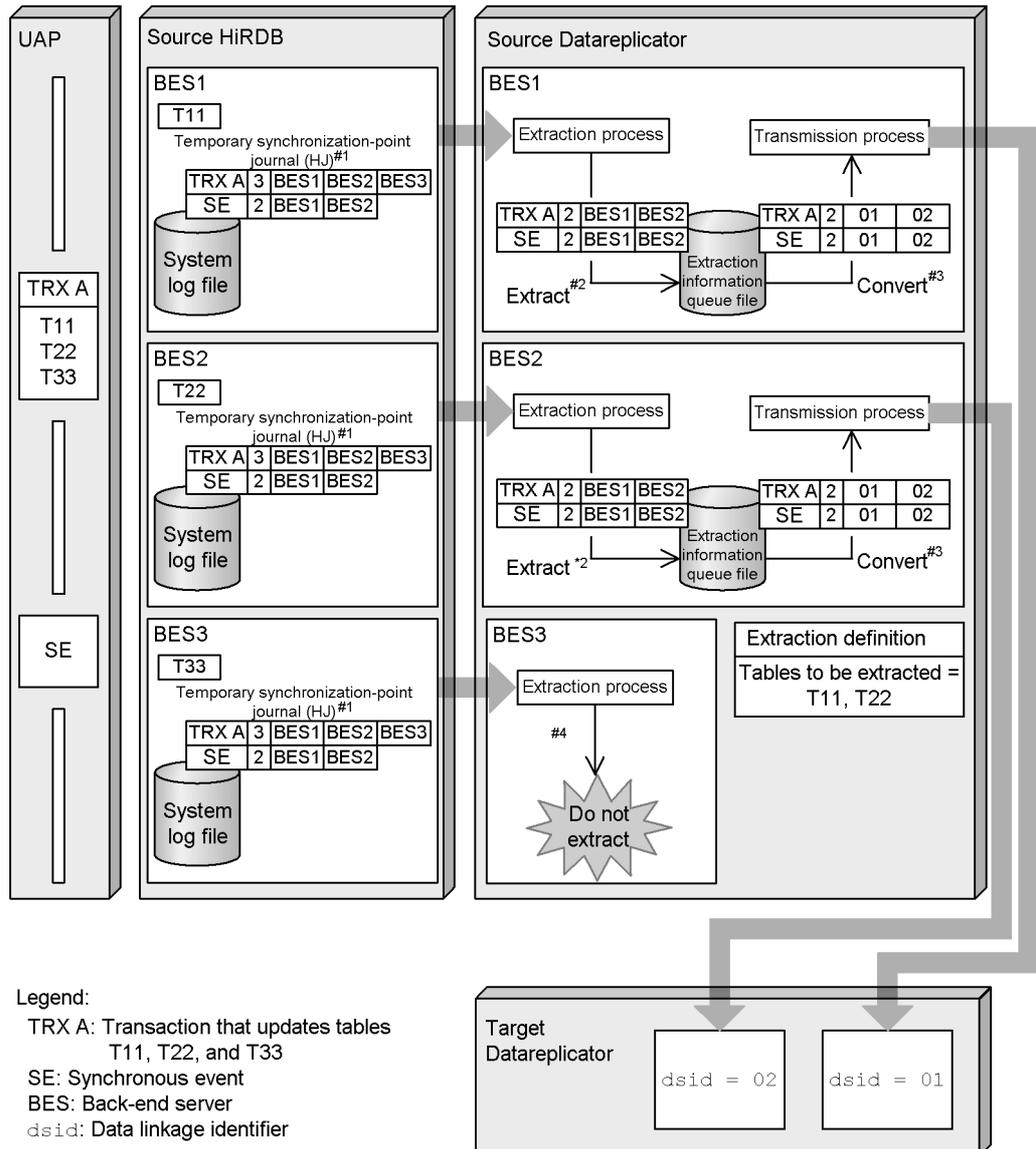
```

3.7.4 Transaction branch information

Transaction branch information is the information about back-end server names that are subject to replication. When the import transaction synchronization facility is used, the transaction branch information is stored in the source HiRDB's system log file. The source Datareplicator extracts that information and guarantees data integrity.

The following figure shows the linkage of transaction branch information.

Figure 3-46: Linkage of transaction branch information



#1

When a transaction is executed on the source HiRDB, transaction branch information is output to HiRDB's system log file for each transaction. At that point, the transaction branch information contains the names of all target back-end servers for the corresponding transaction.

#2

If transaction branch information has been output to the system log file, the source Datareplicator's extraction process unconditionally extracts the transaction branch information and stores it in the extraction information queue file. At this point, the names of the back-end servers that will update a table other than the tables to be extracted are deleted.

#3

The source Datareplicator's transmission process converts the back-end server names in the transaction branch information to the data linkage identifiers, and sends them to the target Datareplicator.

#4

Transaction branch information is not extracted for a back-end server that updates a table that is not the target of extraction.

HiRDB's normal termination log is replaced with a synchronous event and is then transmitted. This enables an import transaction to be synchronized even when normal termination of HiRDB is detected. Note that the synchronous event is extracted when HiRDB starts normally. The import transaction is not synchronized when HiRDB terminates normally.

3.7.5 Synchronous import group

To use the import transaction synchronization facility, you must set the target Datareplicator's data linkage identifiers as a *synchronous import group* in order to use a single target Datareplicator to receive the transaction that is transmitted by each back-end server of the source Datareplicator.

(1) Starting a synchronous import group

To start a synchronous import group, specify the required operands correctly, and then enter one of the following commands:

1. `hdsstart`
2. `hdsrftcl -g synchronous-import-group-name -m start`

The command shown in 2 can be executed only when the synchronization managing process is inactive. If you execute this command while the synchronization managing process is running, the `KFRB00161-E` message is output and the command is ignored. You can use the `hdsstate` command to check whether the process is running.

To start a synchronous import group in disabled status, enter the following command:

```
hdsstart -c synchronous-import-group-name
```

Once you enter this command, the next time the command shown in 1 or 2 is entered, the synchronous import group is started in disabled status. To enable the synchronous

import group again, you must initialize the target Datareplicator.

Note:

A synchronous import group cannot be started for an individual data linkage identifier that constitutes the group. If the `hdsrftcl -d` command is executed on a data linkage identifier constituting a synchronous import group, the `KFRB03304-E` message is output and the command is ignored.

(2) Terminating a synchronous import group

To terminate a synchronous import group, enter one of the following commands:

- `hdsstop#`
- `hdsrftcl -g synchronous-import-group-name -m immediate`

#

For details about the available options, see the `hdsstop` command in Chapter 7. *Command Syntax*.

The following table describes the termination method for each command.

Table 3-20: How a synchronous import group is terminated

Command specification	Termination details
<code>hdsstop</code>	The reception process has stopped and the end of the import information queue file has been detected. If a transaction was executing when the end of the import information queue file was detected, this command rolls back that transaction, and then terminates the synchronous import group.
<code>hdsstop -t event</code>	
<code>hdsstop -q wait-time</code>	
<code>hdsstop -t immediate</code>	Completion of synchronization at the first synchronous event after acceptance of the command, or when the end of the import information queue file is detected. If a transaction was executed when the end of the import information queue file was detected, this command rolls back that transaction, and then terminates the synchronous import group.
<code>hdsrftcl -g synchronous-import-group-name -m immediate</code>	
<code>hdsstop -t force</code>	Rolls back the import transaction that was executing when this command was accepted, and then terminates the synchronous import group. If no import transaction has occurred, the command terminates the synchronous import group as is.

Note:

A synchronous import group cannot be terminated for an individual data linkage identifier that constitutes the group. If the `hdsrfctl -d` command is executed on a data linkage identifier constituting a synchronous import group, the `KFRB03304-E` message is output and the command is ignored.

(3) Notes about synchronous import groups

The notes in this subsection apply to synchronous import groups.

(a) Notes about operand specification

There are limitations on the operand specifications for the data linkage identifiers specified for a synchronous import group, as described in the following table.

Table 3-21: Limitations on the data linkage identifiers specified for a synchronous import group

Definition file name	Operand	Remarks
Import system definition	<code>commit_wait_time</code>	Ignored
	<code>commitment_method</code>	Specify <code>fxa_sqlc</code> . Otherwise, the <code>KFRB00847-E</code> message is output, resulting in an error.
	<code>discintvl</code>	Ignored
Import environment definition	<code>eventtrn</code>	Ignored
	<code>eventtbl</code>	Ignored
	<code>eventretrn</code>	Ignored
	<code>eventrettbl</code>	Ignored
	<code>eventspd</code>	Ignored
	<code>eventcntreset</code>	Ignored
	<code>cmtintvl</code>	Ignored
	<code>trncmtintvl</code>	Ignored
	<code>tblcmtintvl</code>	Ignored
	<code>reflect_trn_max_sqlnum</code>	Ignored
	<code>startmode[#]</code>	Specify <code>trn</code> . Otherwise, the <code>KFRB00847-E</code> message is output, resulting in an error.
<code>ujcodekind</code>	Specify <code>rcv</code> . Otherwise, the <code>KFRB00847-E</code> message is output, resulting in an error.	

Definition file name	Operand	Remarks
Import definition	load statement	by 'uoc-name' cannot be specified. If it is specified, the KFRB03315-E message is output, resulting in an error.

#

Use the `startmode` operand to specify the method for importing the data linkage identifiers specified in a synchronous import group. The import method cannot be changed by the event issuing facility or the `hdsrftl` command.

You must evaluate the specification values beforehand, because starting a synchronous import group in disabled status enables all the parameters, including those that had been ignored.

(b) Notes about the event issuing facility

Among the events for a synchronous import group, only synchronous events are available. An attempt to use any other type of event results in output of the KFRB03316-W message, and the attempt is ignored.

A connection-based termination event is also ignored; it will not be treated as a -1 event.

(c) Notes about errors in a synchronous import group

If there is an error in any of the data linkage identifiers that constitute a synchronous import group, all the data linkage identifiers constituting that synchronous import group terminate with an error.

When an error occurs in a synchronous import group, the synchronization managing process is terminated and each import process that detected termination of the synchronization managing process outputs the KFRB03311-E message.

Termination of a synchronization managing process is detected at one of the following times:

- When or before an SQL statement is issued
- When the value specified in the `ref_wait_interval` operand in the import environment definition elapses after termination of the import information queue file was detected

(d) Notes about the number of transactions that is displayed by the `hdsstate` command

The number of transactions displayed for each data linkage identifier by the `hdsstate` command is the number of transaction branches. It does not match the number of actual global transactions that have occurred at the source system.

3.7.6 How to check the extraction and import status

The method of checking the extraction and import status depends on whether the import transaction synchronization facility is used. The following table shows how to check the extraction and import status.

Table 3-22: How to check the extraction and import status

Purpose	When this facility is not used	When this facility is used
To determine how much of the source database's update information has been imported	At the end of a job, execute the <code>hdeevent</code> command to check the <code>KFRB03211-I</code> message that is output at the target system, in order to make sure that the update information has been imported through the last job.	Use the method described in 3.7.9 <i>Examples</i> .
To determine whether all log information has been extracted from the system log	After executing the <code>pdlogsync -d sys</code> command, execute the <code>pdls -d rpl -j</code> command and make sure that (1) below is greater than (2) below: <pre> SYSTEMID : HRD1(183346) Data replication : Y UNITID : unt1(183346) Data replication : Y SERVER NAME : sds01 Extract Database : Y Extract Status : C System Log Extract Point : Run ID Group Gen No. BLock No. 41418a08 log5 2 4e ... (1) System Log Sync Info : Run ID Group Gen No. BLock No. 41418a08 log5 2 4c ... (2) </pre>	Same as when this facility is not used.
To terminate the source Datareplicator after sending all log information from the extraction information queue file	Place the table subject to extraction in unupdatable status, and then enter the <code>hdestate</code> command. Next, make sure that the <code>read position</code> and <code>write position</code> match in all back-end servers, and then execute the <code>hdestop</code> command.	Place the table subject to extraction in unupdatable status, and then execute a synchronous event. Next, make sure that the <code>read position</code> and <code>write position</code> match in all back-end servers, and then execute the <code>hdestop</code> command.

Purpose	When this facility is not used	When this facility is used
To terminate the import Datareplicator after importing all log information from the import information queue file	Enter the <code>hdsstate</code> command while the source Datareplicator is terminated, make sure that each data linkage identifier's <code>read position</code> and <code>write position</code> match, and then execute the <code>hdsstop</code> command.	<ol style="list-style-type: none"> Place the table subject to extraction in unupdatable status, and then execute a synchronous event. Next, make sure that the <code>read position</code> and <code>write position</code> match in all back-end servers, and then execute the <code>hdestop</code> command to terminate the source Datareplicator. In the status of 1, execute the <code>hdsstop</code> command.

3.7.7 Error handling

The following table lists the errors that might occur while you are using the import transaction synchronization facility and describes the action to be taken for each error.

Table 3-23: Error handling when the import transaction synchronization facility is used

Error	Action
Synchronization was not achieved because the <code>hdevent</code> command was not executed, resulting in the extraction information queue file or import information queue file becoming full.	Disable the synchronous import group.
The extraction information queue file has become full for a reason such as a large transaction.	Use the data linkage recovery facility to recover data.
Update information and synchronous events for some of the data linkage identifiers were lost for a reason such as a disk failure.	Use the data linkage recovery facility to recover data.
A synchronous event for some back-end servers cannot be received due to a communication failure.	Wait until the communication failure is recovered, and then receive the remainder.
A synchronous event cannot be received because the import information queue file is full.	Disable the synchronous import group, restart the target system, and create free space in the import information queue file. When all log information has been imported, re-create the synchronous import group.

Error	Action
Import processing cannot be performed due to a shortage of resources in the target database or due to a timeout.	Take one of the following actions: <ul style="list-style-type: none"> • Change the database settings, and then re-execute. • Disable the synchronous import group, restart the target system, and create free space in the import information queue file. When all log information has been imported, re-create the synchronous import group.
Synchronization cannot be achieved due to an import error on at least one of the data linkage identifiers (such as an SQL error).	Take one of the following actions: <ul style="list-style-type: none"> • Eliminate the cause of the error, and then re-execute. • Skip the erroneous update information by using import suppression or the <code>skip_sqlcode</code> operand in the import environment definition.
A synchronous event or a sequence was lost.	Disable the synchronous import group.

Note:

If you use the error recovery facility in the source system, the import transaction synchronization facility cannot be used because the transaction branch information is not extracted. To recover an error, either delete the synchronous import group definition or start the system while the synchronous import group is disabled.

3.7.8 Notes

This subsection provides notes about using the import transaction synchronization facility.

(1) Limitations on UOCs

The import transaction synchronization facility does not support creation of an import information editing UOC.

(2) Partial initialization of data linkage identifiers

Partial initialization is not supported for just some of the data linkage identifiers that constitute a synchronous import group. If partial initialization is executed on a data linkage identifier constituting a synchronous import group, the `KFRB04344-E` message is output and initialization processing is cancelled. However, if either of the following conditions is satisfied, the entire synchronous import group is initialized:

- The synchronous import group consists of only the one data linkage identifier.
- Other than the data linkage identifier for which partial initialization is specified, no data linkage identifier has been defined or all the other data linkage identifiers are missing numbers.

In the example shown below, the KFRB04344-E message is not output. The situation is treated as initialization of the target Datareplicator.

Command specification

This example starts the target Datareplicator with initialization of dsid003 specified.

```
hdsstart -i -D 003
```

Import system definition

Other than dsid003, all data linkage identifiers are missing numbers:

```
dsid001 = **  
dsid002 = **  
dsid003 = c1  
dsid004 = **  
dsid005 = **  
syncgroup001 = Grp001,c1
```

(3) Large transactions

If the synchronous event for each back-end server is not synchronized during the import transaction synchronization facility processing, the transaction becomes large and a resource shortage might occur at the target HiRDB. If a large transaction cannot be completed, resulting in rollback, a large amount of rollback log information is output.

To avoid such a problem, specify the maximum number of transactions to wait for until synchronization. Make this specification in the `syncwait_limit_tran_count` operand of the import system definition in such a manner that the target HiRDB's lock pool size and system log file size are not exceeded.

(4) Deleting all rows (PURGE TABLE)

When you execute `PURGE TABLE`, data linkage is supported if the source HiRDB is a parallel server and the table is not row-partitioned among multiple servers.

Notes:

If `PURGE TABLE` is executed, an import transaction is committed automatically. Therefore, the following actions must be taken while no update transaction has been executed on the source database:

1. Execute a synchronous event to achieve synchronization and obtain the status where no other transaction has been executed.
2. Execute `PURGE TABLE`. The `PURGE TABLE` execution conditions are as follows:

- There is only one `PURGE TABLE` in the transaction to be executed.
- There is neither `INSERT`, nor `UPDATE`, nor `DELETE` before `PURGE TABLE`.

3. Re-execute the synchronous event.

If any other operation is performed, only the import transaction with the data linkage identifier on which `PURGE TABLE` was executed is committed, and the import transaction's synchronization with other data linkage identifiers can no longer be guaranteed. If this occurs, you must execute a synchronous event immediately to synchronize the import transaction.

(5) Changing the extraction definition

To change the extraction definition, you must initialize the target Datareplicator. For details about how to initialize the target Datareplicator, see *6.8.4 Changing the configuration when the import transaction synchronization facility is used*.

If you change the extraction definition without initializing the target Datareplicator, the `KFRB03317-E` message is output during execution of the target Datareplicator, resulting in an error. To avoid this error, you must disable the synchronous import group, and then restart the system.

(6) HiRDB client environment definition in the target Datareplicator execution environment

If you have specified `USER` in the `HDSCLTWAITTIME` environment variable for the target Datareplicator, specify a value of 0 for `PDSWAITTIME` and `PDSWATCHTIME` in the HiRDB client environment definition. If any other value is specified, the import SQL process might result in a connection timeout.

3.7.9 Examples

This subsection describes examples of operations using the import transaction synchronization facility.

The following table describes the procedure for each operation:

Operation	Procedure
Initialization procedure at the environment configuration stage	Follow the procedure described in <i>6.2 Initialization procedure at the environment configuration stage</i> .
Normal job	Follow the procedure described in <i>Starting through terminating a normal job</i> .
<ul style="list-style-type: none"> • Changing the configuration of table to be extracted • Changing the server configuration for the source HiRDB 	Follow the procedure described in <i>Starting through terminating a normal job</i> , and then change the extraction and import definitions. Then follow the procedure described in <i>6.8.4 Changing the configuration when the import transaction synchronization facility is used</i> .

3. Data Linkage Facilities

Operation	Procedure
Partial initialization of the source Datareplicator	Partial initialization is not supported for a destination that uses the import transaction synchronization facility. Follow the procedure described in <i>6.8.4 Changing the configuration when the import transaction synchronization facility is used.</i>

Starting through terminating a normal job

The following figure shows the procedure for starting through terminating a normal job.

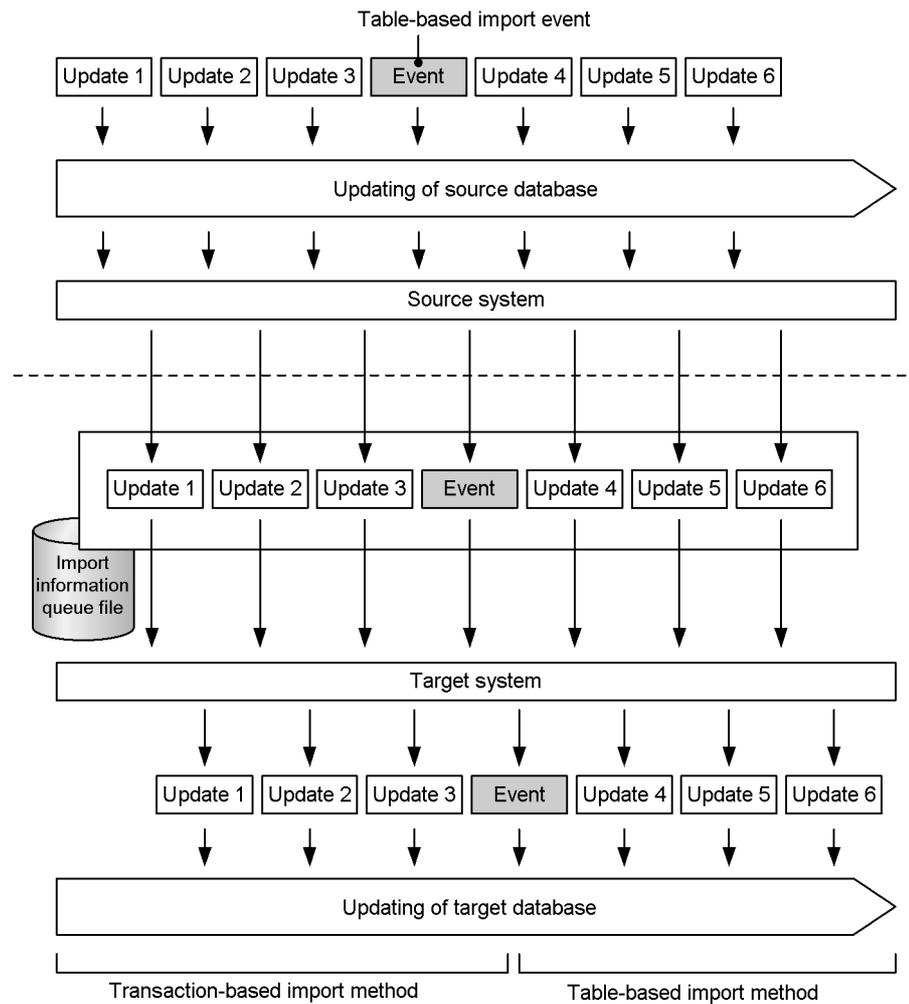
1. Starting the target Datareplicator	
Execute the command at the target system.	
Command to be executed	<code>hdsstart</code>
Check item	Check that the <code>KFRB00100-I</code> message has been output to the import error information file.
2. Starting the source Datareplicator	
Execute the command at the source system.	
Command to be executed	<code>hdestart</code>
Check item	Check that the <code>KFRB00502-I</code> message has been output to the extraction master error information file.
3. Starting a job	
4. Issuing a synchronous event periodically	
Execute the <code>hdeevent</code> command periodically to issue a synchronous event.	
5. Terminating the job	
Terminate the job for the source database.	
6. Terminating the source Datareplicator	
Execute the command at the source system.	
Command to be executed	<code>hdeevent -n <i>synchronous-event-code</i></code> <code>hdeevent -n 0</code>
Check item	Check that the <code>KFRB02032-I</code> message has been output to the extraction node master error information file and that the message shows <code>status = Event</code> .
Command to be executed	<code>hdestop</code>
Check item	Check that the <code>KFRB00503-I</code> message has been output to the extraction master error information file.
7. Terminating the target Datareplicator	
Execute the command at the target system.	
Command to be executed	<code>hdsstop</code>
Check item	Check that the <code>KFRB00104-I</code> message has been output to the import error information file.

3.8 Event facility

The event facility switches import operations on the basis of events issued at the source system. The event facility enables you to stop import processing, restart import processing, and change the import method for the active import processing.

The following figure shows an example of operations when the event facility is used.

Figure 3-47: Example of operations when the event facility is used



Note Updates 1-3 are imported using the transaction-based import method. If the source system issues a table-based import event before update 4 occurs, the table-based import method becomes effective for updates 4-6.

3.8.1 Issuing events

When you execute the `hdevent` command on the source Datareplicator, an event is issued. Specify in the `hdevent` command an event code that you specified in the transmission environment definition or the import environment definition. For details about defining event codes, see *3.8.3 Defining event codes*.

If you will be using the event facility, create an event control table. If you use a UAP to update the event control table, events can also be issued from the UAP. For details about the event control table, see *4.6.7 Designing the event control table*.

3.8.2 Types of events

The following table lists and describes the types of events.

Table 3-24: Types of events

Type of event	Description
Event to reset the data-transmission count	Event that resets the source Datareplicator's data transmission count (the count that is displayed as <code>Send data Transmission count</code> in the execution result of the <code>hdstate</code> command).
Synchronous event	Event that first synchronizes an import transaction between back-end servers, and then commits. This event is applicable only when the import transaction synchronization facility is used.
Transaction-based import event	Event that requests import processing in units of transactions.
Table-based import event	Event that requests import processing in units of tables.
Transaction-based import restart event	Event that restarts inactive import processing in units of transactions.
Table-based import restart event	Event that restarts inactive import processing in units of tables.
Stop event	Event that requests termination of import processing.
Event to reset the import processing count	Event that resets the number of import processes executed (the count displayed as <code>Reflection count</code> in the execution result of the <code>hdsstate</code> command).
Connection-based termination event	Event that is sent from the source system when the source system completes transfer of update information. The code of the connection-based termination event is always <code>-1</code> (fixed; you cannot specify this event code in the import environment definition).

3.8.3 Defining event codes

Each event is identified by an event code. The following table lists the operands that define event codes.

Table 3-25: Operands for defining event codes

Type of event	Event code (default)	Operand name	Type of definition
Event to reset the data-transmission count	--	<code>eventcntreset</code>	Transmission environment definition

Type of event	Event code (default)	Operand name	Type of definition
Synchronous event	--	eventsync	
Transaction-based import event	1	eventtrn	Import environment definition
Table-based import event	2	eventtbl	
Transaction-based import restart event	3	eventretrn	
Table-based import restart event	4	eventretbl	
Stop event	5	eventspd	
Event to reset the import processing count	--	eventcntreset	
Connection-based termination event	-1 (fixed)	(cannot be specified)	

Legend:

--: There is no default value for the event code.

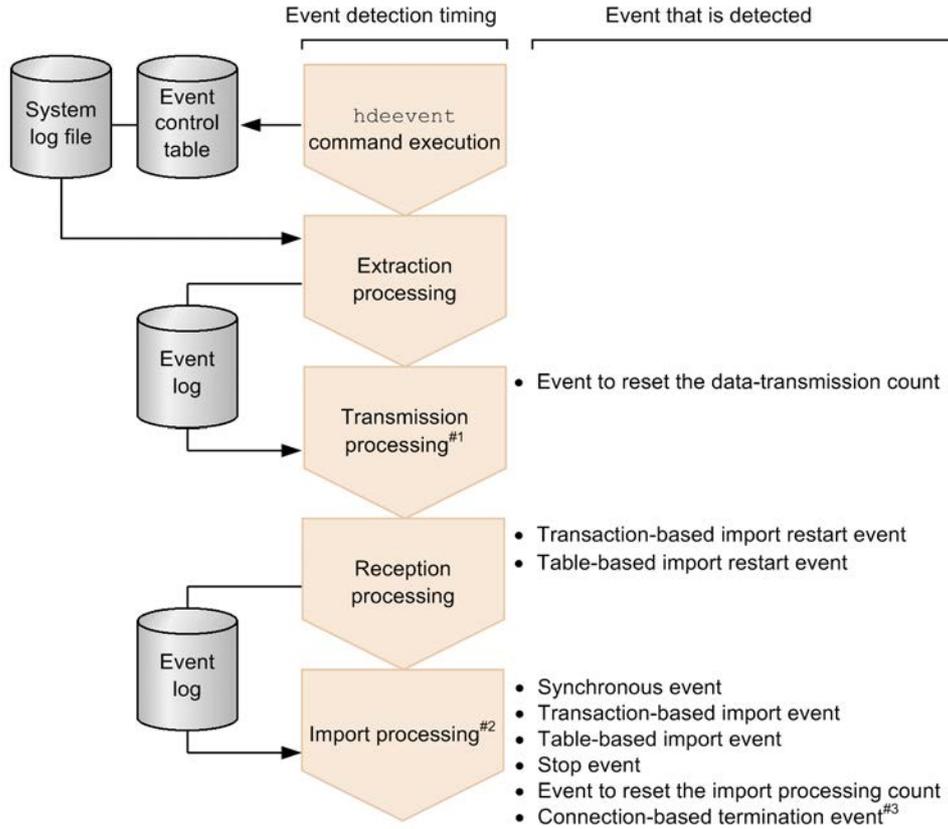
3.8.4 Event detection timing and action

This subsection explains the event detection timing and the action taken by the extraction or import facility when an event is detected.

(1) *Event detection timing*

The following figure shows the event detection timing and the import processing switching timing.

Figure 3-48: Event detection timing



#1

The KFRB02036-I message is output.

#2

The KFRB03006-I or KFRB03211-I message is output.

#3

Detected by the hdsstop command execution.

(2) Action when an event is detected

When an event is detected, the extraction or import facility executes a corresponding process. The following table explains the actions of the extraction or import facility's processes for the various events.

Table 3-26: Extraction or import facility's actions for event processes

Type of event	Type of process					
	TX process	Import master process	RX process	Import definition server process	Import process	Synchronization managing process
Event to reset the data-transmission count	Resets the transmission data count and resumes processing.	--	--	--	--	--
Synchronous event	--	--	--	--	Commits all import processes that belong to the same synchronous import group.	Synchronizes import processes that belong to the synchronous import group.
Transaction-based import event	--	--	--	Restarts the import process in <code>trn</code> mode.	Outputs a message and terminates the process (termination mode: event termination).	--
Table-based import event	--	--	--	Restarts the import process in <code>tbl</code> mode.	Outputs a message and terminates the process (termination mode: event termination).	--

3. Data Linkage Facilities

Type of event	Type of process					
	TX process	Import master process	RX process	Import definition server process	Import process	Synchronization managing process
Transaction-based import restart event	--	Restarts the import definition server process (if it is already active, outputs an error message).	Reports to the import master process.	Restarts the import process in the <code>trn</code> mode, and then starts all import processes by detecting this event code, outputs a message, and resumes processing.	Outputs a message and resumes processing.	--
Table-based import restart event	--	Restarts the import definition server process (if it is already active, outputs an error message).	Reports to the import master process.	Restarts the import process in the <code>tbl</code> mode, and then starts all import processes by detecting this event code, outputs a message, and resumes processing.	Outputs a message and resumes processing.	--
Stop event	--	--	--	Waits for termination of events for all import processes; when all import processes have been terminated by events, outputs a message and terminates processing.	Outputs a message and terminates the process (termination mode: event termination).	--

Type of event	Type of process					
	TX process	Import master process	RX process	Import definition server process	Import process	Synchronization managing process
Event to reset the import processing count	--	--	--	--	Resets the import processing count and resumes processing.	--
Connection-based termination event	--	--	--	When all import processes detect this event code, outputs a message and resumes processing.	Outputs a message and resumes processing.	--

Legend:

TX: Transmission

RX: Reception

--: No action is taken.

trn mode: Transaction-based import method

tbl mode: Table-based import method

Note:

When an event is detected, Datareplicator executes synchronization point processing on the target database, regardless of the type of event.

3.8.5 Notes

The following notes apply to using the event facility:

- If the event code sent from the source system is not a code specified in the import environment definition or is not -1 (connection-based termination event), the target Datareplicator recognizes that an event is being reported and outputs a message, but it continues the import processing that was underway before the event was notified (the event is ignored).
- If import processing is already underway, Datareplicator ignores a restart event

(transaction-based import restart event or table-based import restart event), if detected.

- If you terminate Datareplicator before processing a received restart event (transaction-based import restart event or table-based import restart event), information about the received restart event is discarded.
- The import definition server process monitors events at an interval of one second and can manage up to 16 events concurrently. If more than 16 events are issued within one second, the additional event codes will not be displayed in messages. When you use event codes in messages output by the import definition server process, be sure that 16 events per second is not exceeded.

3.9 Duplexing files

Datareplicator enables you to duplex files for data linkage, thereby improving reliability in the event of file errors.

Note:

The time required for data linkage processing doubles when duplexed files are used because data is written to two files. When you are determining whether to use file duplexing, take into account this increase in data linkage processing time.

When you duplex files, define for each file type a desired logical file in the environment definition, and then define in the duplexing definition file the two physical files that are to constitute the logical file. One of these files is called the *primary file* and the other is called the *secondary file*. The contents of the primary and secondary files are the same. The following figure shows the relationship between a logical file and the primary and secondary physical files.

System	Definition file
Target system	Import master status file
	Import status file
	Import information queue file

Notes:

- The physical files must be allocated to the same file type (character special files or regular files).
- All physical files must have the same sector length.
- For a data linkage file, specify a physical file name identical to the logical file name as either the primary or secondary file name.
- File duplexing is not applicable to files that use the Datareplicator file system area. Make sure that a duplexed logical file is not allocated in the file system area.

3.9.2 Files associated with duplexing

The following table describes the files that are associated with file duplexing.

Table 3-28: Files associated with duplexing

File	Purpose
Extraction system definition file or import system definition file	Defines the name of the duplexing definition file.
Extraction environment definition file or import environment definition file	Defines the name of the logical files that are to be duplexed.
Duplexing definition file	Defines the names of the physical files for each file that is to be duplexed.
Duplexing control file	Controls file duplexing. Datareplicator creates this file during initialization processing.

(1) Extraction system definition file and import system definition file

These files specify the name of the duplexing definition file.

To duplex files for the source system, define the `file_dupenv` operand in the extraction system definition file; to duplex files for the target system, define the `file_dupenv` operand in the import system definition file. For details about the `file_dupenv` operand, see 5.2 *Extraction system definition* and 5.8 *Import system definition*. If you omit this operand, you cannot use file duplexing.

(2) Extraction environment definition file and import environment definition file

These files specify the names of the logical files to be duplexed.

To duplex files for the source system, define the names of the logical files in the extraction environment definition file; to duplex files for the target system, define the names of the logical files in the import environment definition file.

(3) Duplexing definition file

This file defines the correspondence between the names of logical and physical files. For details about the definitions, see *5.7 Duplexing definition (source)* or *5.12 Duplexing definition (target)*.

(4) Duplexing control file

This file controls file duplexing. Datareplicator creates the duplexing control file in the following directories during initialization processing:

System	Directory	File name
Source	Source Datareplicator directory at all HiRDB server machines	hde_fileenv.prp
Target	Target Datareplicator directory	hds_fileenv.prp

After initialization of Datareplicator, make a backup copy of the duplexing control file. If an error occurs on this file, you will have to use its backup file to restore the file.

3.9.3 Notes about duplexing**(1) When the system switchover facility is used**

- After Datareplicator initialization, copy the duplexing control file to the Datareplicator directory of the corresponding standby server. There is no need to copy the duplexing control file in the following cases:
 - A HiRDB/Parallel Server is used with a system switchover configuration and the executing back-end server is located on the standby host.
The duplexing control file is created automatically on the back-end server in the executing system.
 - There is no change to the duplexing control file during the second or a subsequent system switchover processing.
 - There is no need to copy to the standby system files whose extension is .mf. These are work files Datareplicator creates when duplex files are used. Do not delete these files while Datareplicator is running.

(2) In the event of a file error

If an error occurs in a duplexed file, the file must be recovered. For details, see *9.1.6*

Handling of file errors during file-duplexed operation.

Chapter

4. System Design

This chapter explains the procedures for designing a data linkage system appropriate to the application mode, as well as the procedures for designing the source and target Datareplicators.

- 4.1 System design items
- 4.2 Designing a linkage pattern
- 4.3 Designing the correspondence between source and target databases
- 4.4 Designing the correspondence between source and target systems
- 4.5 Designing the data linkage system mode
- 4.6 Designing a source Datareplicator
- 4.7 Designing a target Datareplicator
- 4.8 Designing the source HiRDB

4.1 System design items

The following table lists and describes the design items, together with their related definitions and commands.

Table 4-1: Design items and their related definitions and commands

Design item	Description	Related definition and command#
Linkage pattern	Design the pattern of data linkage.	Source system definition: <ul style="list-style-type: none"> • Extraction definition Target system definition: <ul style="list-style-type: none"> • Import definition
Correspondence between source and target databases	Design the correspondence between the source and target databases subject to data linkage, such as mapping keys, data types, and character code set. If the target table contains no data, the data must be created using a program such as the HiRDB database load utility or HiRDB Dataextractor.	N/A
Correspondence between source and target systems	Design the correspondence between the source and target systems, such as the proportions and the data linkage identifiers required to establish correspondence.	Source system definitions: <ul style="list-style-type: none"> • Extraction system definition • Extraction environment definition • Transmission environment definition • Extraction definition Target system definitions: <ul style="list-style-type: none"> • Import system definition • Import environment definition
Data linkage system mode	Design the data linkage system as either a unidirectional or bidirectional update system and its application to the hierarchical system.	Source system definitions: <ul style="list-style-type: none"> • Extraction system definition • Extraction environment definition • Transmission environment definition • Extraction definition Target system definitions: <ul style="list-style-type: none"> • Import system definition • Import environment definition

Design item	Description	Related definition and command [#]
Source Datareplicator [#]	Design the correspondences among files, the extraction/transmission method, the startup and termination methods, and so on for the source Datareplicator.	Source system definitions: <ul style="list-style-type: none"> • Extraction system definition • Extraction environment definition • Transmission environment definition • Commands: <ul style="list-style-type: none"> - hdestart - hdestop - hdeevent
Target Datareplicator [#]	Design the correspondences among files, the import method, the startup and termination methods, and so on for the target Datareplicator.	Target system definitions: <ul style="list-style-type: none"> • Import system definition • Import environment definition • Commands: <ul style="list-style-type: none"> - hdsstart - hdsstop - hdsrftcl

[#]: If the source or target system is a mainframe system (XDM/SD E2, XDM/RD E2, ADM, PDM2 E2, or TMS-4V/SP), define the information in XDM/DS. For details about the XDM/DS definitions and commands, see the manual *VOS3 XDM Data Linkage Facility XDM/DS Description and Definition* and the manual *VOS3 XDM Data Linkage Facility XDM/DS Operation*. If the source system is a mainframe database that uses SAM files (PDM2 E2 or RDB1 E2), see the applicable product's manual.

4.2 Designing a linkage pattern

There are seven data linkage patterns for Datareplicator. You select the appropriate pattern for your data linkage needs; you can also combine multiple patterns.

The following table lists and describes the data linkage patterns.

Table 4-2: Data linkage patterns

Data linkage pattern	Description
Data linkage to a table with the same format	Imports a source table into an identical target table.
	Imports a source table by changing only the table name and/or column names.
Data linkage to a table with a different format	Sorts a source table's columns at the source, and then extracts and imports them.
	Sorts a source table's columns at the target, and then imports them.
	Selects some of the source table's columns at the source, and then extracts and imports them.
	Selects some of the source table's columns at the target, and then imports them.
	Adds fixed-value columns to a source table at the target, and then imports the table.
Data linkage from one table to n tables	Divides a source table into multiple tables and extracts them at the source, and then imports them into a single target.
	Divides a source table into multiple tables and extracts them at the source, and then imports them into multiple targets.
	Divides a source table into multiple tables at the target, and then imports them.
Data linkage from n tables to one table	Combines multiple source tables at the target, and then imports them into a single table.
Data linkage for import of selected rows	Selects rows to be transmitted, and then imports them.
Data linkage using a UOC routine	Uses a UOC routine to edit update information at the target, and then imports it.
Acquisition of a record of update information over time	Acquires information such as the import dates and times as well as the update information in chronological order.

This section explains the source system's extraction definition for each data linkage

pattern. The explanations provided here are based on the Datareplicator definitions in the source system.

If the source database is a mainframe database (XDM/SD E2, XDM/RD E2, ADM, PDM2 E2, or TMS-4V/SP), you must use XDM/DS to specify the extraction definition. For details about how to use XDM/DS for the extraction definition, see the *VOS3 XDM/DS* manual. If the source database is a mainframe database that uses SAM files (PDM2 E2 or RDB1 E2), some definition specifications might not be required, depending on the product; for details, see the applicable product's manual.

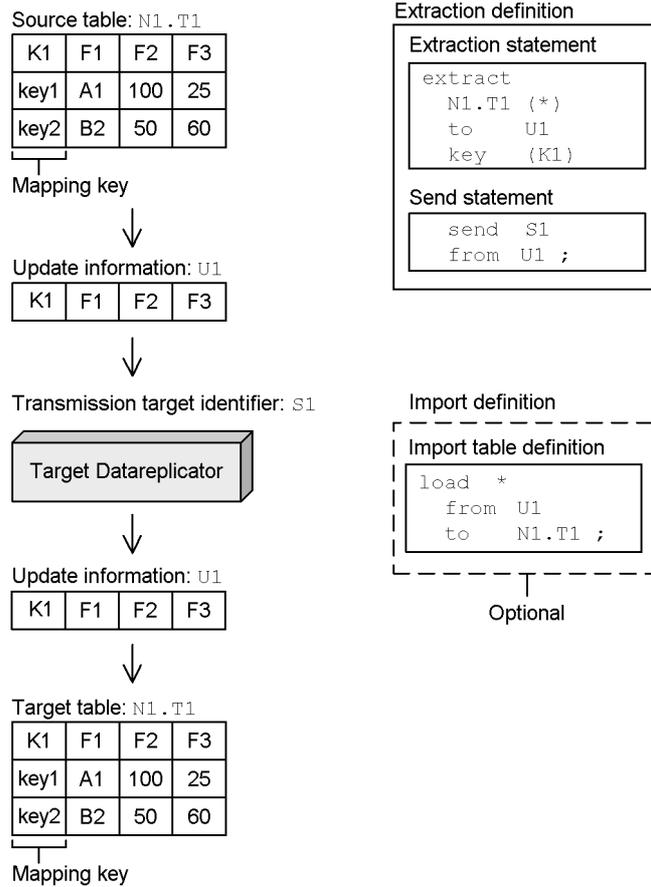
4.2.1 Data linkage to a table with the same format

This section explains the pattern for data linkage to a table with the same format.

(1) *Example of importing a source table into an identical table*

The following figure shows an example of importing a source table into an identical table that has the same table format, table name, column names, and so on.

Figure 4-1: Example of importing into an identical table



Explanation

- Tables subject to processing

At the source, use the extraction statement in the extraction definition to specify the source table, in *authorization-identifier.table-identifier* format.

At the target, use the `to` clause of the import table definition to specify the target table, in *authorization-identifier.table-identifier* format.

- Columns to be extracted

At the source, use the extraction statement to specify the columns to be extracted. To extract all columns without sorting them, you can use the asterisk (*) in place of the column names.

- Name of the update information

Define with a single extraction statement an extraction condition for the single table. Use the `to` clause of the extraction statement to define a name for the update information that is to be extracted based on this extraction condition; this name is called the *update information name*. To specify multiple extraction conditions for a single table, specify one extraction statement for each extraction condition.

- Destination of the extracted update information

Specify with the `send` statement in the extraction definition the destination of the extracted update information. Use the `send` statement to specify one update information name for each destination. To specify multiple update information names for a single destination, specify one `send` statement for each update information name that is to be sent.

- Mapping keys

Mapping keys are used to identify the rows to be extracted or imported. You must establish the correspondence of the mapping keys between the source and target tables. You use the `key` clause of the extraction statement to specify the mapping key.

- Omitting the import definition

You can omit the import definition in this example because it imports a table into an identical table (same table and column names). When you omit the import definition, Datareplicator uses the extraction definition for the import definition and executes data linkage. Note that the assumed authorization identifier, table identifier, and column names depend on the source database. The following table shows the values that are assumed for these items when the import definition is omitted.

Table 4-3: Values assumed for the authorization identifier, table identifier, and column names when the import definition is omitted

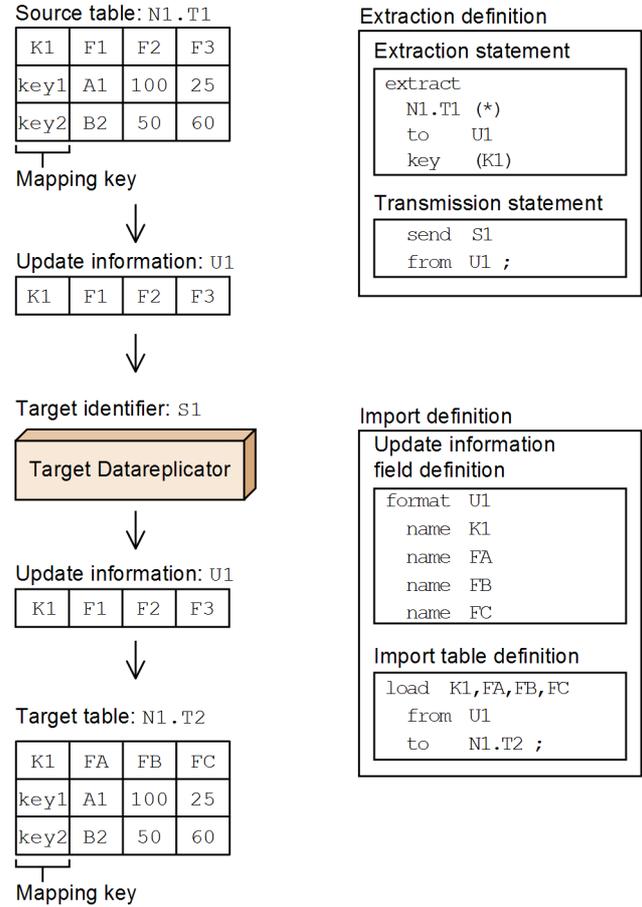
Source database							Target database
HiRDB	XDM/SD E2#	XDM/RD E2	ADM	PDM 2 E2	TMS-4V/ SP	RDB1 E2	HiRDB
Authorization identifier	Schema name	Authorization identifier	Database name	DBM name	Authorization identifier	Authorization identifier	Authorization identifier
Table identifier	Least significant record type name	Table identifier	Segment	Dataset name	Table identifier	Table identifier	Table identifier
Column name	Component name	Column name	Field name	Field name	Column name	Column name	Column name

#: If the component name and the redefined item name specified with the RESTRUCT statement in the extraction redefinition are not unique within the same extraction definition, you cannot omit the import definition because the import table corresponding to the redefined item name cannot be defined.

(2) Example of importing a source table by changing the table name or column names

The following figure shows an example of importing a source table by changing the table name or column names.

Figure 4-2: Example of importing by changing the table name or column names



Explanation

- Changing the column names definition
Define the new column names in name clauses in the update information field

definition. You must define in these clauses all field names in the update information. Of the field names defined here, specify in the `load` statement in the import table definition those fields that are subject to import processing. If you are importing all fields in the same sort order, you can omit the update information field definition.

- Changing the table name or column names

Specify the new column names in the `load` statement in the import table definition and specify the new table name in the `to` clause.

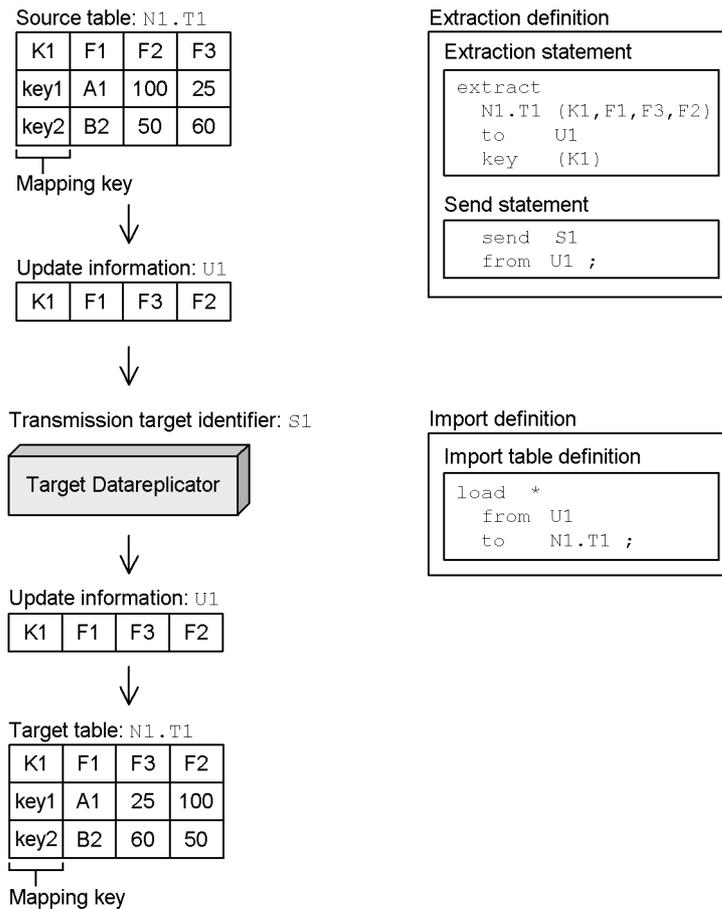
4.2.2 Data linkage to a table with a different format

This section explains the pattern for data linkage to a table with a different format.

(1) Example of sorting a source table's columns at the source, and then extracting and importing them

The following figure shows an example of sorting a source table's columns at the source, and then extracting and importing them.

Figure 4-3: Example of sorting columns at the source, and then extracting and importing them



Explanation

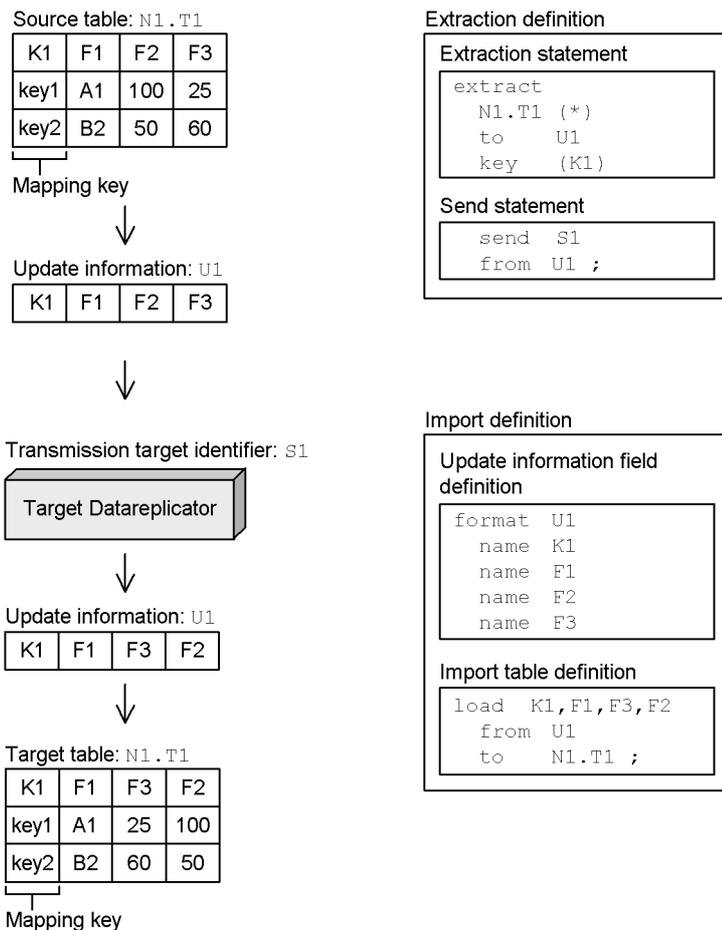
- Sorting columns at the source

To sort columns at the source, you use the extraction statement to specify the source table's column names in the order they are to appear (the re-sorted order).

(2) Example of sorting a source table's columns at the target, and then importing them

The following figure shows an example of sorting a source table's columns at the target, and then importing them.

Figure 4-4: Example of sorting columns at the target, and then importing them



Explanation

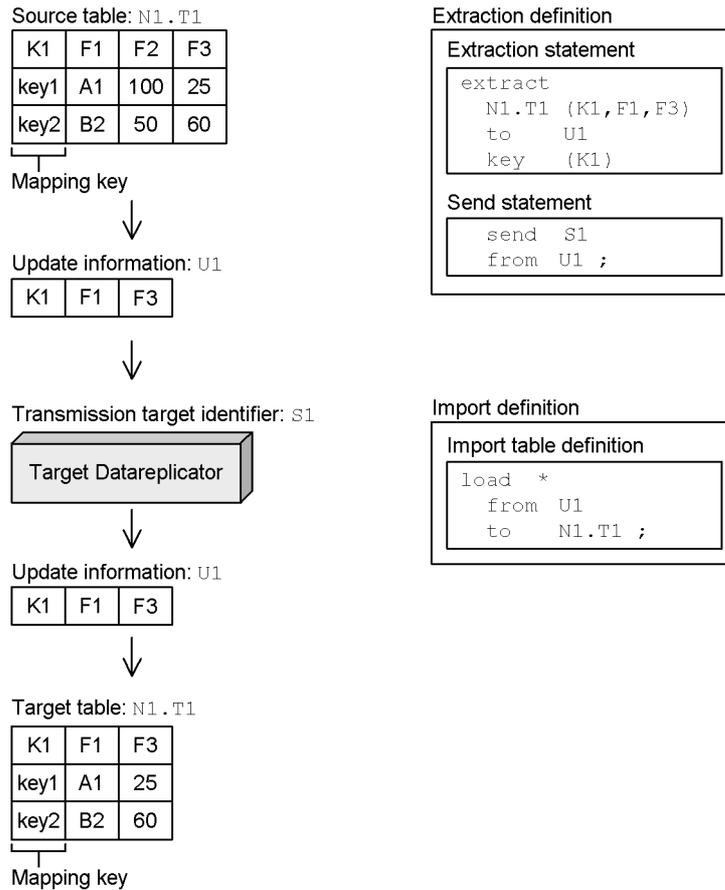
- Sorting columns at the target

To sort columns at the target, first use name clauses in the update information field definition to define the field names of the update information, and then use the `load` statement in the import table definition to specify these fields in the order they are to appear (the re-sorted order).

(3) Example of selecting some of the source table's columns at the source, and then extracting and importing them

The following figure shows an example of selecting some of the source table's columns at the source, and then extracting and importing them.

Figure 4-5: Example of selecting some of the columns at the source, and then extracting and importing them



Explanation

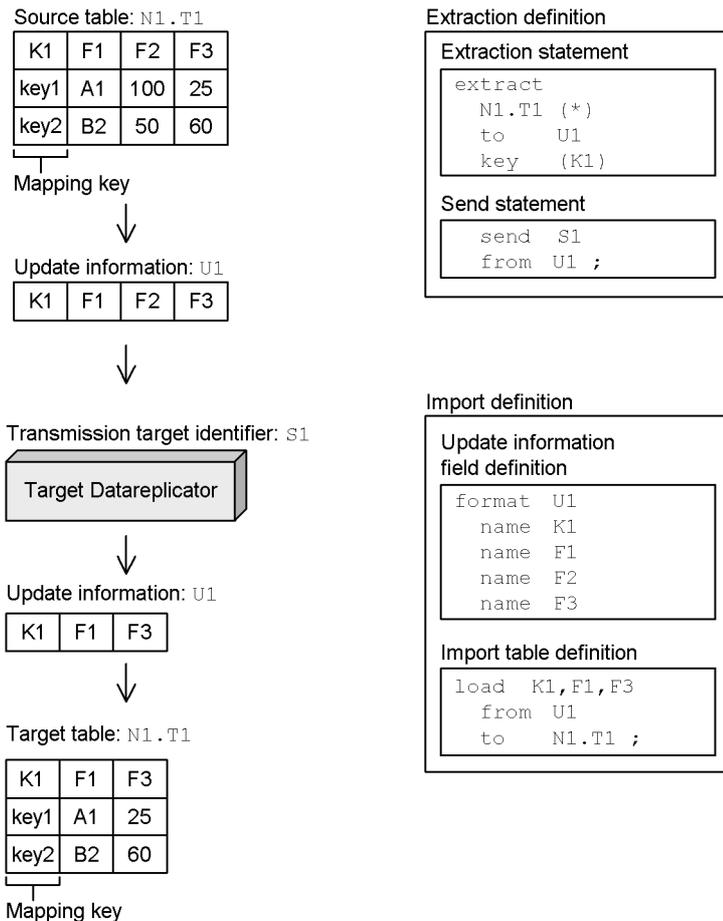
- Selecting columns at the source

To select some of the columns at the source and extract them, specify in the extraction statement the columns that are to be extracted.

(4) Example of selecting some of the source table's columns at the target, and then importing them

The following figure shows an example of selecting some of the source table's columns at the target, and then importing them.

Figure 4-6: Example of selecting some of the columns at the target, and then importing them



Explanation

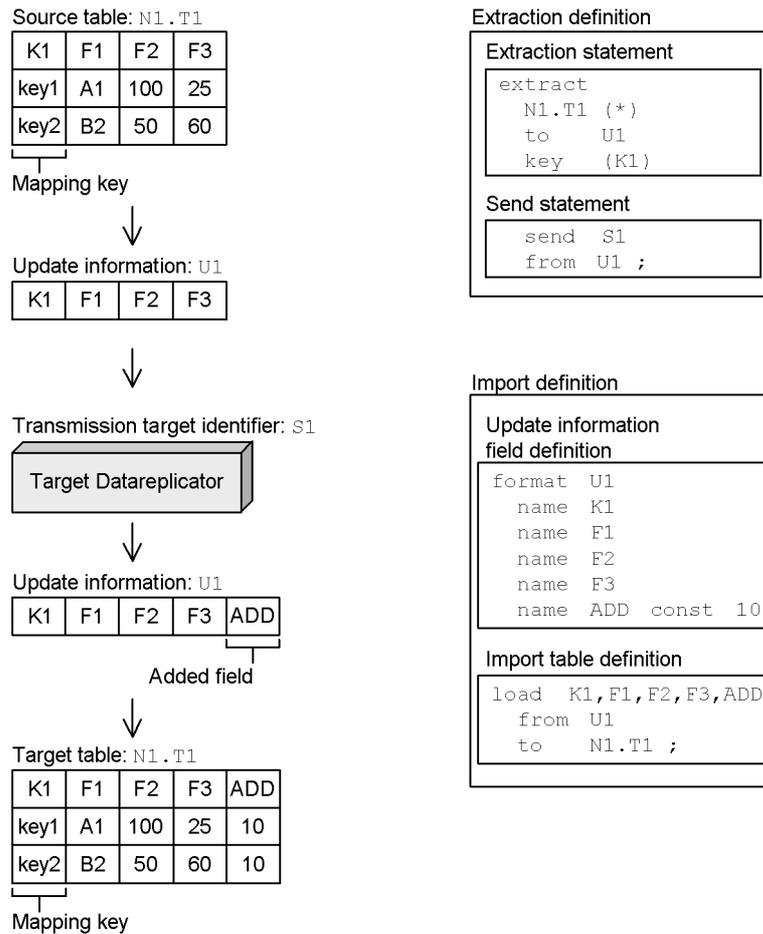
- Selecting columns at the target

To select some of the columns at the target and import them, first use name clauses in the update information field definition to define the field names of the update information, and then use the load statement in the import table definition to specify only those fields that are to be imported.

(5) Example of adding fixed-value columns to a source table at the target, and then importing them

The following figure shows an example of adding fixed-value columns to a source table at the target, and then importing them.

Figure 4-7: Example of adding fixed-value columns at the target, and then importing them



Explanation

- Field names of update information when adding a fixed-value column

To add a fixed-value column, select a field name and specify it in the update information field definition. Following the specified field name, specify `const` and the fixed value that is to be set in the field. You must also specify this user-defined field name in the `load` statement in the import table definition.

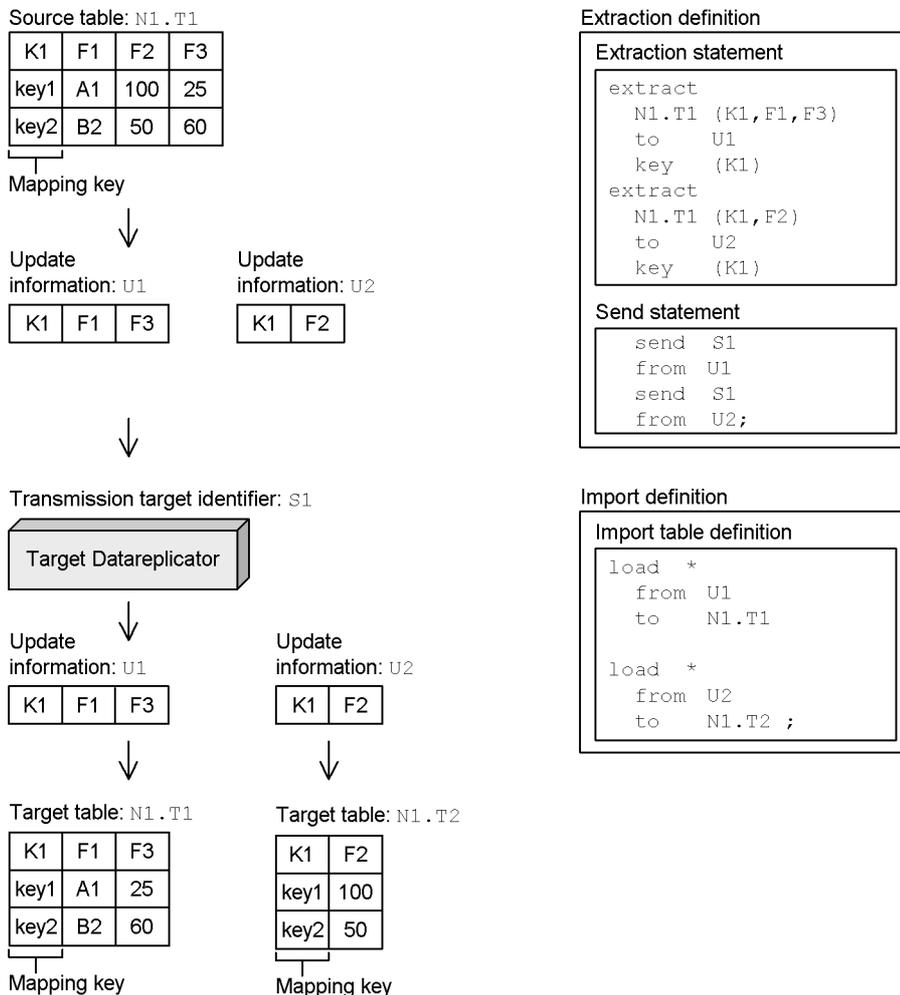
4.2.3 Data linkage from one table to *n* tables

This section explains the pattern for data linkage from one table to *n* tables.

(1) Example of dividing a source table into multiple tables and extracting them at the source, and then importing them into a single target

The following figure shows an example of dividing a source table into multiple tables and extracting them at the source, and then importing them into a single target.

Figure 4-8: Example of dividing a table into n tables and extracting them at the source, and then importing them into a single target system



Explanation

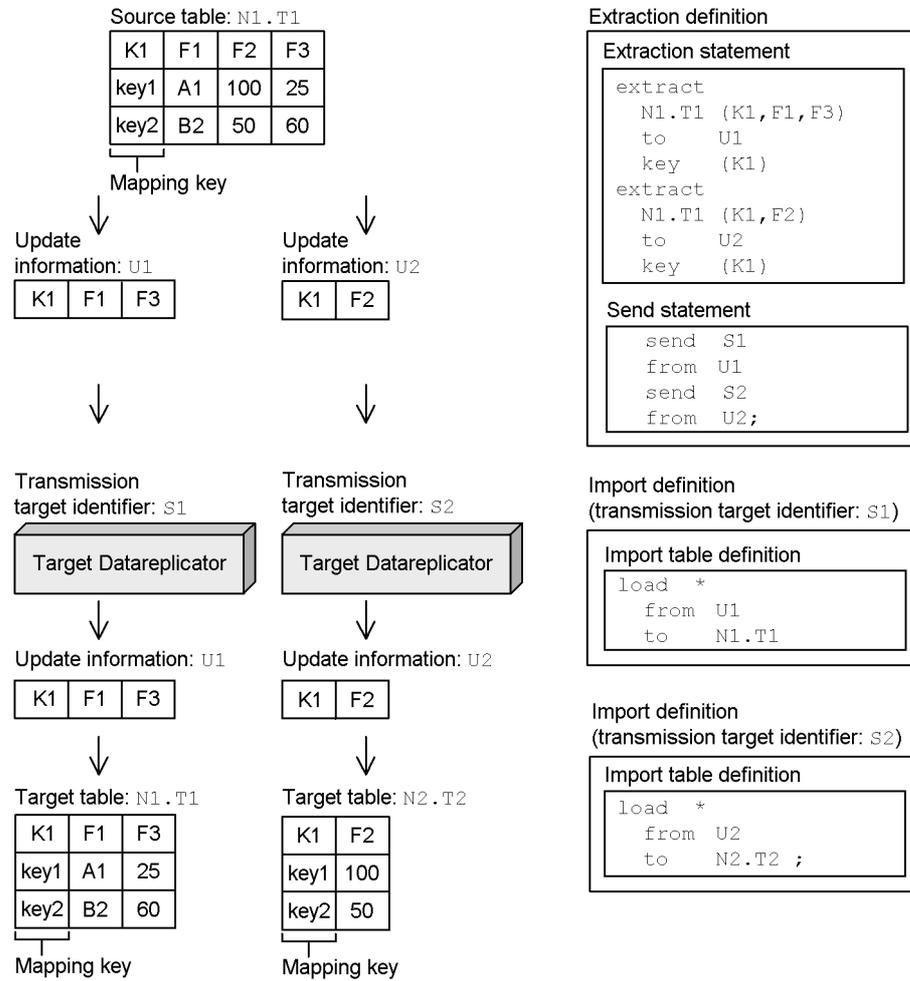
- Dividing a table at the source
 To divide a table into multiple tables at the source, specify as many extraction

statements as there are divisions of the table.

(2) Example of dividing a source table into multiple tables and extracting them at the source, and then importing them into multiple targets

The following figure shows an example of dividing a source table into multiple tables and extracting them at the source, and then importing them into multiple targets.

Figure 4-9: Example of dividing a source table into multiple tables and extracting them at the source, and then importing them into multiple targets



Explanation

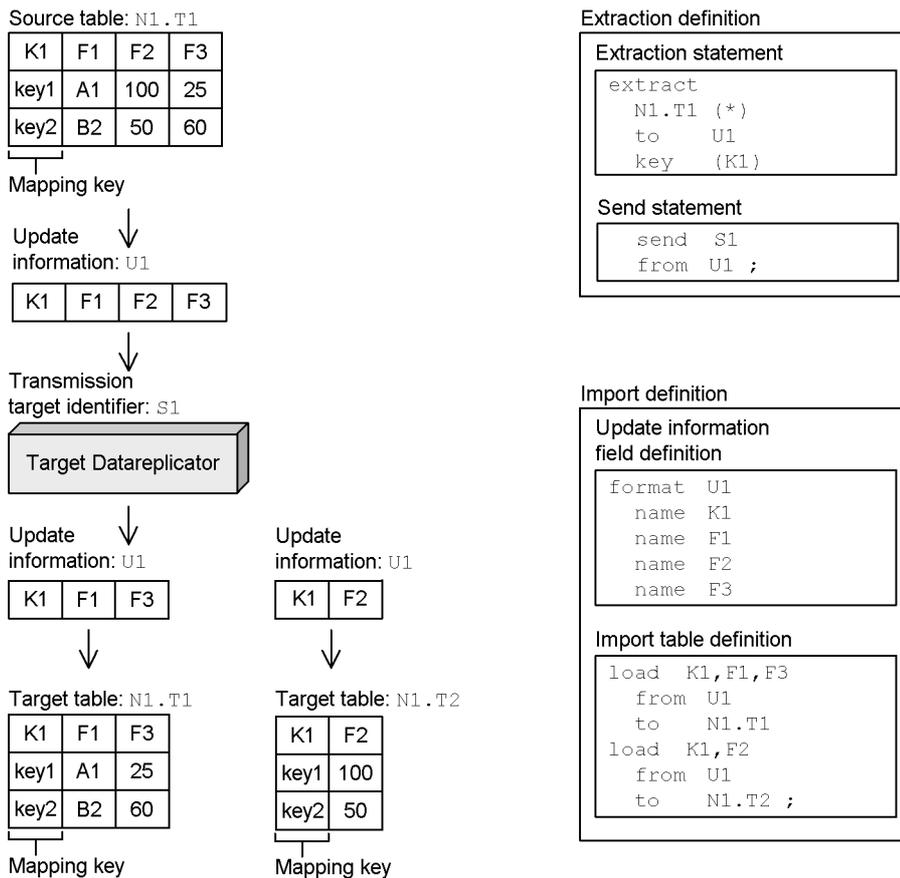
- Sending to multiple targets

To send a table to multiple targets, specify as many send statements as there are targets.

(3) Example of dividing a source table into multiple tables at the target, and then importing them

The following figure shows an example of dividing a source table into multiple tables at the target, and then importing them.

Figure 4-10: Example of dividing a source table into multiple tables at the target, and then importing them



Explanation

- Dividing a table at the target

To divide a source table into multiple tables at the target, specify as many import table definitions as there are divided tables.

Because mapping keys are needed to establish the correspondence between the source and target tables, specify the mapping key fields in all import table definitions.

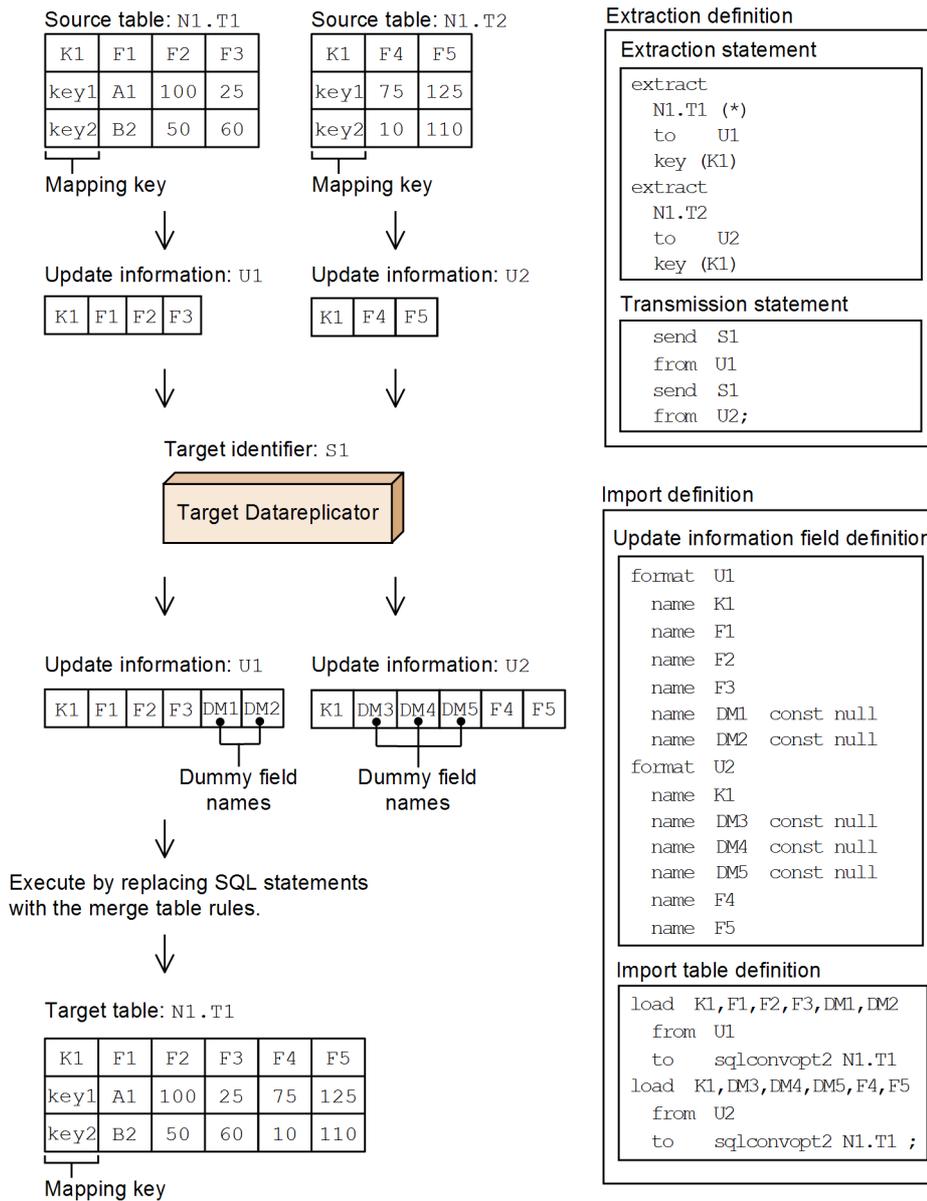
4.2.4 Data linkage from n tables to one table

This section explains the pattern for data linkage from n tables to one table.

You use a *merge table* to combine multiple source tables into one at the target for import into a single table. You can create a merge table at the target system, combine multiple source tables into one table, and then import it into a single table. Changes made by such commands as `INSERT`, `DELETE`, and `PURGE TABLE` are converted according to the merge table rules or the `load` statements specified in the import table definition.

The following figure shows an example of combining multiple source tables into one table at the target, and then importing that table into a single table using a merge table.

Figure 4-11: Example of combining multiple source tables into one at the target, and then importing it into a single table using a merge table



Explanation

- Extracting and sending multiple tables at the source

To extract update information for multiple tables at the source, specify as many extraction statements as there are tables to be extracted.

- Importing multiple tables into a single table at the target

To import multiple tables into a single table at the target, specify as many update information field definitions and import table definitions as there are tables to be extracted. In this case, specify user-defined dummy field names in the update information field definitions because correspondence must be established between the numbers of columns at the source and target tables. Use a `const` clause to specify each field name.

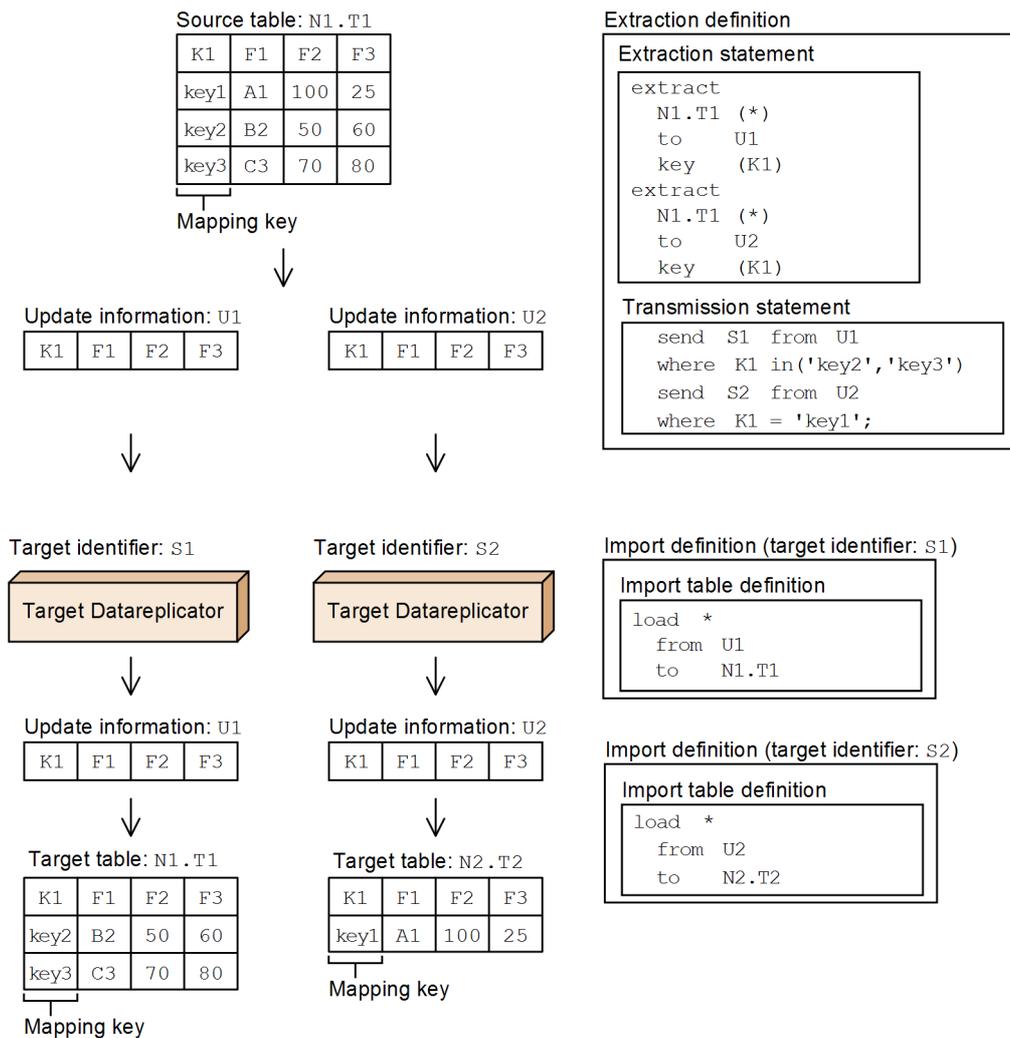
You must specify in the import table definitions all the field names that are specified in the update information field definitions. When you create a merge table, specify `sqlconvopt1` or `sqlconvopt2` in front of the table name in the `to` clause in the import table definition.

4.2.5 Data linkage by selecting rows to be sent

This section explains the pattern for data linkage that selects from the source system that portion of the update information that satisfies specified conditions, and then imports it. You can specify a condition for the update information to be imported into each import system.

The following figure shows an example of linking data by selecting rows to be sent.

Figure 4-12: Example of linking data by selecting rows to be sent



Explanation

- Specifications for the source Datareplicator
Specify the selection conditions for the mapping key with send statements in the source Datareplicator's extraction definitions.
- Sending to multiple target systems
You can send update information conditionally to multiple target systems by specifying as many send statements as there are target systems.

4.2.6 Data linkage using a UOC routine

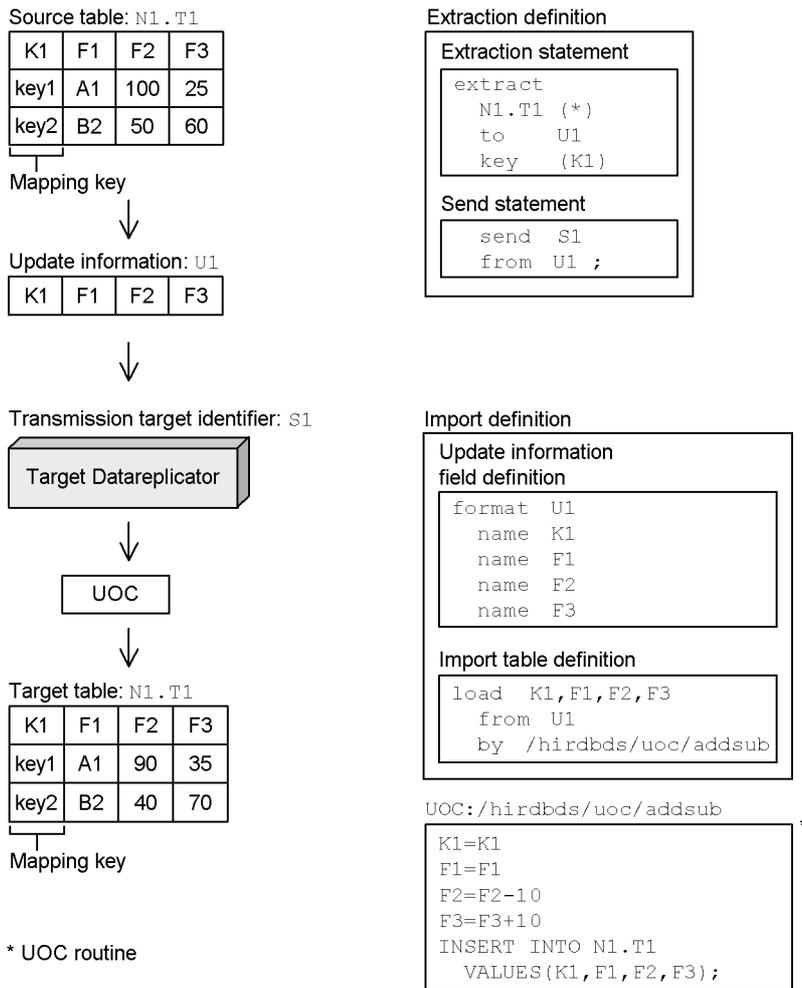
This section explains the pattern for data linkage that uses a UOC routine to edit data, and then import it. For details about UOC routines, see Chapter 8. *User Own Coding Routines*.

(1) Example of using a UOC routine to edit update information at the target, and then importing it

You can use a UOC routine to edit and process update information before importing it.

The following figure shows an example of using a UOC routine to edit update information at the target, and then importing it.

Figure 4-13: Example of using a UOC routine to edit update information at the target, and then importing it



Explanation

- Name of UOC routine used to edit update information
To use a UOC routine to edit update information before importing it, specify the name of the UOC routine in the `by` clause of the import table definition.
- UOC routine's processing
The field data in the update information specified in the `load` statement of the import table definition is passed to the UOC routine. The UOC routine must

process the received data, and then create and issue an SQL statement to execute import processing.

Use the functions provided by the target Datareplicator to create the UOC routine, and then compile and link it.

- Sample UOC routines

The target Datareplicator provides sample UOC routines. You can use these routines to create your own UOC routines.

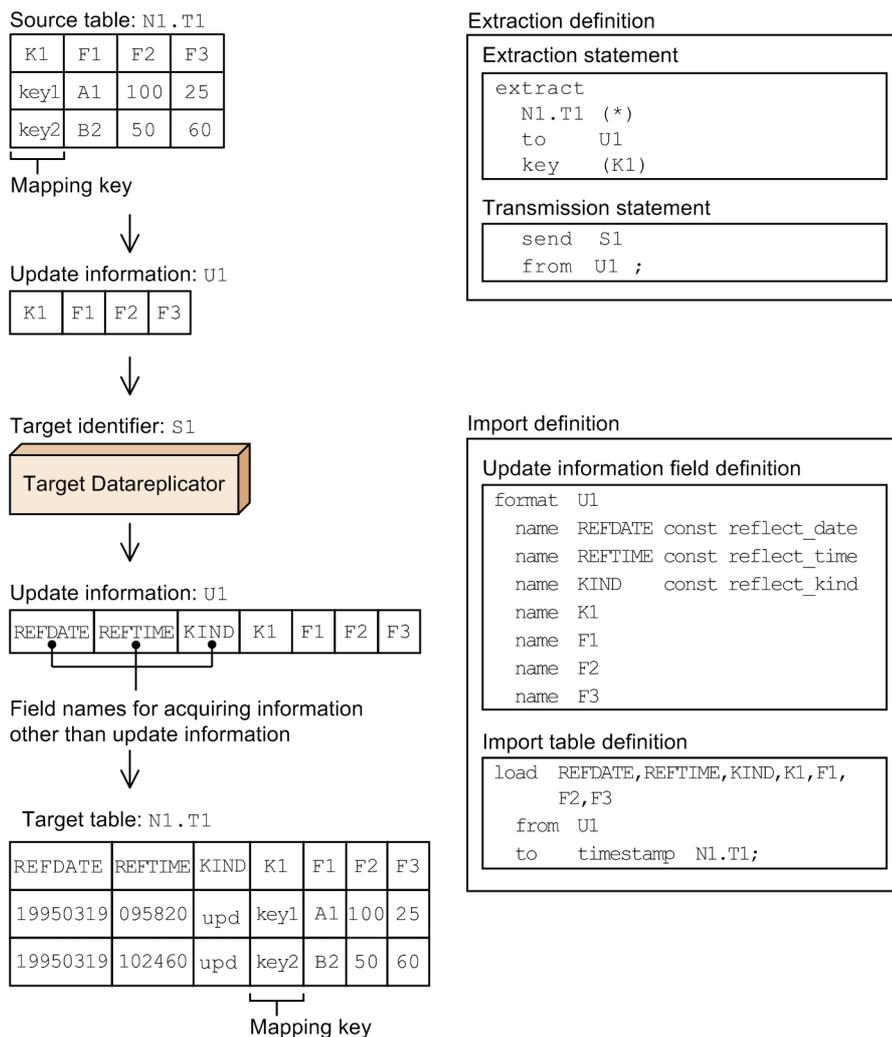
4.2.7 Acquisition of a record of update information over time

This section explains the pattern that acquires a record of update information over time (in chronological order).

(1) Example of acquiring information such as the import dates and times as well as the update information in chronological order

The following figure shows an example of acquiring information such as the import dates and times as well as the update information in chronological order.

Figure 4-14: Example of acquiring information such as the import dates and times as well as the update information in chronological order



Explanation

- Acquiring table data in chronological order
 To acquire data in chronological order, specify `timestamp` before the table name in the `to` clause of the import table definition. To acquire information other than update information, you must create in advance columns for receiving the information in the target table.

4. System Design

- **Information other than update information**

To acquire information other than update information, such as the import dates, times, and types, you must select and specify field names in name clauses in the update information field definition. Following each field name, specify `const` and the literal indicating the information that is to be acquired.

You must specify the names of all fields to be subject to acquisition in the `load` statement of the import table definition.

4.3 Designing the correspondence between source and target databases

This section explains system design for establishing correspondence between source and target databases.

4.3.1 Conditions for the source database

This section explains the conditions for extracting data from a HiRDB database. For details about the conditions for extracting data from a mainframe database (XDM/SD E2, XDM/RD E2, ADM, PDM2 E2, or TMS-4V/SP), see the *VOS3 XDM/DS* manual; for details about the conditions for extracting data from a mainframe database using SAM files (PDM2 E2 or RDB1 E2), see the applicable product manual.

(1) Tables subject to extraction processing

For details about the tables that are subject to extraction processing, see *1.3 Tables supported for data linkage*.

(2) Data types subject to extraction processing

For details about the data types that are subject to extraction processing, see *1.4.1 Data types supported by Datareplicator*.

For details about the correspondence between the source and target databases, see *4.3.4 Designing for the supported data types*.

(3) SQL statements subject to extraction processing

For details about the SQL statements that are subject to extraction processing, see *1.5 SQL statements supported for data linkage*.

(4) UAPs subject to extraction processing

The source Datareplicator executes data linkage on the basis of the database update log in the system log. Therefore, all update processing on a database subject to data linkage by a UAP that outputs database update log information is subject to extraction processing.

When you execute a UAP to update a database that is subject to data linkage while data linkage is in effect, some environment variables and SQL statements require special handling. The following table provides notes on UAP execution.

Table 4-4: Notes on UAP execution

Item	Notes
Specification of no-log mode in an environment variable	Do not specify NO in the PDDBLOG environment variable in HiRDB's client environment definition.

Item	Notes
Modification of definitions by SQL statements	While data linkage is in effect, do not issue any of the following SQL statements for a database that is subject to data linkage: <ul style="list-style-type: none"> • PURGE TABLE (for a partitioned table spanning multiple back-end servers[#]) • ALTER TABLE • DROP TABLE For details about how to modify definitions during execution of data linkage, see 6.5.6 <i>Source HiRDB handling procedure</i> .

#: When the source Datareplicator detects a PURGE TABLE statement for a table that spans multiple back-end servers, it outputs a warning message to the error information file without extracting the corresponding update information. If this happens, you can use the `prg_eventno` operand in the transmission environment definition to terminate transmission processing or to issue an event to the target Datareplicator.

(5) Utilities subject to extraction processing

HiRDB Datareplicator supports only the database load utility (`pdload`) for extraction. Note that if there is an error in specifying options, inconsistency might occur between extraction and import processing. Note also that limitations in Datareplicator preclude its use to extract BLOB data.

If other utilities that output update log information are executed during data linkage, inconsistency occurs between extraction and import processing. To execute other utilities, you must specify the options so as to prevent them from operating within the extraction range.

The following table provides notes on utility execution.

Table 4-5: Notes on utility execution

Utility name	Notes
Database definition utility	<div style="display: flex; align-items: flex-start;"> <div style="flex: 1; padding-right: 10px;"><code>pddef</code></div> <div> While data linkage is in effect, do not issue any of the following SQL statements for a database that is subject to data linkage: <ul style="list-style-type: none"> • ALTER TABLE • DROP TABLE For details about how to modify the definitions during execution of data linkage, see 6.5.6 <i>Source HiRDB handling procedure</i>. </div> </div>

Utility name		Notes
Database load utility	pdload	<ul style="list-style-type: none"> Existing table Do not execute the utility in the creation mode during execution of data linkage (you cannot specify the <code>-d</code> option). Specify the log acquisition mode in the addition mode (specify <code>a</code> in the <code>-l</code> option). New table To add a new table subject to linkage during execution of data linkage, use the log acquisition mode (specify <code>a</code> in the <code>-l</code> option). If you will be applying data linkage to <code>pdload</code>, also see the chapter that includes <i>4.7.3(9)(a) Number of locked resources</i>.
Database structure modification utility	pdmmod	During execution of data linkage, do not re-initialize an RDAREA that contains a table subject to extraction processing.
Database reorganization utility	pdrorg	During execution of data linkage, use the no-log mode or pre-update log acquisition mode (specify <code>n</code> or <code>p</code> in the <code>-l</code> option). If you use the log acquisition mode, conformity is lost between the source and target databases because update information about only a portion of the reorganized database is output to the system log.
Database recovery utility	pdrstr	To recover a database, use the database's backup copy and the log collected since the backup was made. If you attempt to recover a database by any other method, conformity is lost between the source and target databases.
Rebalancing utility	pdrbal	To execute the rebalancing utility on a table subject to data linkage, you must modify the table definition (addition of RDAREAs). For details about how to modify a table definition and an extraction definition, see (3) in <i>6.5.6 Source HiRDB handling procedure</i> . Database update processing by the rebalancing utility is not subject to extraction processing.

Note:

If a table subject to utility processing contains a BLOB column, the rows updated by the utility will not be imported into the target database because the source Datareplicator ignores database updating by a utility.

(6) Amount of update information subject to extraction processing

The maximum amount of update information that can be extracted by the source Datareplicator (sum of the definition lengths of columns subject to extraction) is 256 megabytes.

If the amount of update information exceeds 256 megabytes, the extraction definition preprocessor command results in an error. Note that SGMLTEXT-type and XML-type columns are not subject to checking. To extract SGMLTEXT-type and XML-type columns, you must take into account the data lengths handled as these types so that the columns to be extracted do not exceed 256 megabytes.

4.3.2 Creating a table subject to import processing

This section lists the procedures for creating a table that will be subject to import processing.

(1) *Creating a table subject to import processing*

A table that is subject to import processing must have already been created. There are three ways to create such a table:

- With HiRDB's database load utility
- With HiRDB's SQL statements
- With HiRDB Dataextractor

For details about how to create a table with HiRDB, see the *HiRDB Version 9 System Operation Guide*. For details about how to create a table using HiRDB Dataextractor, see the *HiRDB Dataextractor Version 8* manual.

(2) *Conditions for a table that is to be subject to import processing*

A table you create must satisfy the following conditions:

- It must be a base table.
- A column for storing mapping keys must be defined.
- The number of columns in the table cannot exceed 4,000.

4.3.3 Designing the correspondence of mapping keys

A *mapping key* is used to identify the correspondence between update information and rows in the target table. Therefore, you must establish the correspondence between the mapping keys in the source and target tables. A mapping key consists of one or more columns.

Note 1

A mapping key is used to identify target rows that correspond to source data. If the correspondence between the mapping keys in the source and target cannot be established, unintended rows might become subject to processing, introducing problems with the remaining import processing.

When a specific row cannot be identified by the mapping key:

Datareplicator might terminate with the `row not found` error during data import. In such a case, specify the `SQLCODE` to be skipped in the `skip_sqlcode` operand in the import environment definition, and then re-execute.

Note 2

For the target table, define a unique key index that consists of all mapping key

columns. If no unique key index is defined, the Datareplicator cannot detect duplicate key errors, and the import performance of UPDATE and DELETE drops considerably.

The procedure for specifying a mapping key depends on the source database. The following explains how to specify a mapping key for each type of source database.

(1) Mapping key when the source database is HiRDB

The following table lists the data types that can be specified as mapping keys.

Table 4-6: Data types that can be specified as mapping keys (when the source database is HiRDB)

Classification	Data type that can be specified as mapping key	Column definition length
Numeric type	INTEGER	--
	SMALLINT	--
	DECIMAL(m, n)	$1 \leq m \leq 38, 0 \leq n \leq 38, n \leq m$
	FLOAT	--
	SMALLFLT	--
Character type	CHAR(n)	$n \leq 255$
	VARCHAR(n)	$n \leq 255$
	NCHAR(n)	$n \leq 127$
	NVARCHAR(n)	$n \leq 127$
	MCHAR(n)	$n \leq 255$
	MVARCHAR(n)	$n \leq 255$
Date type	DATE	--
	TIME	--
	TIMESTAMP(p)	$p = 0, 2, 4, 6$
	INTERVAL YEAR TO DAY	--
	INTERVAL HOUR TO SECOND	--

Legend:

--: Not applicable

Note:

A repetition key cannot be specified as a mapping key.

- Specify a mapping key in the `key` or `ukey` clause of the extraction statement in the extraction definition.
- You can specify up to 16 columns for one mapping key.
- Once you specify a column in the `key` clause of an extraction definition statement in the extraction definition, you must not update the mapping key. If you update the mapping key, the target database will be updated based on the new key value; as a result, the wrong row might be updated in the target database, and conformity between the source and target databases might be lost. To update a mapping key, specify the applicable column in the `ukey` clause.
- The data corresponding to all mapping key rows that are manipulated by `INSERT`, `UPDATE`, and `DELETE` is sent to the target Datareplicator.

(2) Mapping key when the source database is XDM/SD E2

- For details about how to specify a mapping key when the source database is XDM/SD E2, see the *VOS3 XDM/DS* manual.
- The following data is sent to the target Datareplicator:

During `ERASE` of a record subject to extraction processing

The record subject to extraction processing and the data corresponding to the mapping key at the higher levels

During `ERASE` of a record at a higher level than the record subject to extraction processing

The deleted record and the data corresponding to the mapping key at the higher levels

During `STORE` or `MODIFY` of a record subject to extraction processing

The record subject to extraction processing and the data corresponding to the mapping key at the higher levels

(3) Mapping key when the source database is XDM/RD E2

- For details about how to specify a mapping key when the source database is XDM/RD E2, see the *VOS3 XDM/DS* manual.
- The data corresponding to all mapping key rows that are manipulated by `INSERT`, `UPDATE`, and `DELETE` is sent to the target Datareplicator.

(4) Mapping key when the source database is ADM

- For details about how to specify a mapping key when the source database is ADM, see the *VOS3 XDM/DS* manual.

- The following data is sent to the target Datareplicator:

Data corresponding to the mapping keys for `ISRT`, `REPL`, and `DELT` calls

(5) Mapping key when the source database is PDM2 E2

(a) Data linkage using XDM/DS

- For details about how to specify a mapping key when the source database is PDM2 E2, see the *VOS3 XDM/DS* manual.

(b) Data linkage using SAM files

- When the source database is PDM2 E2, specify the mapping key in the update information definition; for details about how to specify the update information definition, see *5.11 Update information definition*.
- The contents of the SAM file transferred from the source database is stored in an import information queue file at the target Datareplicator.

(6) Mapping key when the source database is TMS-4V/SP

- For details about how to specify a mapping key when the source database is TMS-4V/SP, see the *VOS3 XDM/DS* manual.
- The data corresponding to all mapping key rows that are manipulated by `INSERT`, `UPDATE`, and `DELETE` is sent to the target Datareplicator.

(7) Mapping key when the source database is RDB1 E2

- When the source database is RDB1 E2, mapping keys are created by RDB1 E2's update information extraction facility.
- The contents of the SAM file transferred from the source database are stored in an import information queue file at the target Datareplicator.

(8) Extracted columns of the SMALLFLT or FLOAT type

If a mapping key includes a `SMALLFLT` or `FLOAT` column of the round number type, the data format depends on the source system. The round number type might result in `SQLCODE 100` (no row satisfies the conditions) during `SQL` statement execution. For this reason, do not specify in a mapping key a column of the `FLOAT` or `SMALLFLT` type.

4.3.4 Designing for the supported data types

The following table shows the relationships between table data types and the data types that can be imported by Datareplicator:

Table 4-7: Relationships between table data types and the data types that can be imported by Datareplicator

Data type	Data format and length	Support of character encoding conversion
INTEGER	Same as the HiRDB data format and length	N
SMALLINT		N
DECIMAL		N
LARGE DECIMAL		N
FLOAT		N
SMALLFLT		N
CHAR		Y
VARCHAR		Y
LONG VARCHAR	Recognized as VARCHAR	Y
NCHAR	Same as the HiRDB data format and length	Y
NVARCHAR		Y
LONG NVARCHAR	Recognized as NVARCHAR	Y
MCHAR	Same as the HiRDB data format and length	Y
MVARCHAR		Y
LONG MVARCHAR	Recognized as MVARCHAR	Y
DATE	Same as the HiRDB data format and length	N
TIME		N
TIMESTAMP		N
INTERVAL YEAR TO DATE		N
INTERVAL HOUR TO SECOND		N
SGMLTEXT		Y
FREWORD		Y
XML		N
BLOB		N
BINARY		N

Data type	Data format and length	Support of character encoding conversion
Repetition column		Y

Legend:

Y: Character encoding conversion is supported.

N: Character encoding conversion is not supported.

The relationships between source and target data types might vary depending on the source database. Therefore, you must establish the correspondence of data types between the source and target tables; otherwise, import processing might result in an error.

This section explains the data type correspondences for each type of source database.

(1) Correspondence of data types when the source database is HiRDB

The following table shows the correspondence of data types between the source and target tables when the source database is HiRDB.

Table 4-8: Correspondence of data types between the source and target tables when the source database is HiRDB

Data type in source table	Data type in target table
INTEGER	INTEGER
SMALLINT	SMALLINT
DECIMAL	DECIMAL
LARGE DECIMAL	LARGE DECIMAL
FLOAT	FLOAT
SMALLFLT	SMALLFLT ^{#1}
CHAR	CHAR OR MCHAR
VARCHAR	VARCHAR OR MVARCHAR
NCHAR	NCHAR
NVARCHAR	NVARCHAR
MCHAR	CHAR OR MCHAR
MVARCHAR	VARCHAR OR MVARCHAR
DATE	DATE

Data type in source table	Data type in target table
TIME	TIME
TIMESTAMP	TIMESTAMP
INTERVAL YEAR TO DAY	INTERVAL YEAR TO DAY
INTERVAL HOUR TO SECOND	INTERVAL HOUR TO SECOND
SGMLTEXT ^{#2}	SGMLTEXT ^{#2}
FREWORD	FREWORD
XML	XML
BLOB ^{#2}	BLOB ^{#2}
BINARY	BINARY

Notes:

- An error occurs at the target HiRDB if you attempt to import data that is longer than the target table column or whose numeric value overflows.
- You can specify a repetition column. However, data extraction cannot be performed in units of elements in the repetition column.

#1

Data linkage is supported, but rounding errors occur because values are rounded.

#2

To use data linkage, specify ALL in the RECOVERY operand in the HiRDB table definition. For details about HiRDB table definition, see the manual *HiRDB Version 9 SQL Reference*.

(2) Correspondence of data types when the source database is XDM/SD E2

The following table shows the correspondence of data types between the source and target tables when the source database is XDM/SD E2.

Table 4-9: Correspondence of data types between the source and target tables when the source database is XDM/SD E2

Data subject to extraction processing ^{#1}		Data type in target table
Data attribute	Data length	
CHAR(<i>n</i>)	$1 \leq n \leq 30,000$	CHAR(<i>n</i>)

Data subject to extraction processing ^{#1}		Data type in target table
Data attribute	Data length	
CHAR(<i>n</i>)	$n > 30,000$	VARCHAR(<i>n</i>) ^{#3}
NCHAR(<i>n</i>)	$1 \leq n \leq 15,000$	NCHAR(<i>n</i>)
NCHAR(<i>n</i>)	$n > 15,000$	NVARCHAR(<i>n</i>) ^{#3}
PACK(<i>m, n</i>)	--	DECIMAL(<i>m + n, n</i>) ^{#4}
UNPACK(<i>m, n</i>) ^{#2}	--	DECIMAL(<i>m + n, n</i>) ^{#4}
COMP(<i>m, n</i>)	$1 \leq m \leq 4, n = 0$	SMALLINT
COMP(<i>m, n</i>)	$5 \leq m \leq 9, n = 0$	INTEGER
COMP(<i>m, n</i>)	$m = 0$ or $m > 10$ or $n \neq 0$	DECIMAL(<i>29, n</i>) ^{#6}
BINARY(<i>m, n</i>)	$1 \leq m \leq 15, n = 0$	SMALLINT
BINARY(<i>m, n</i>)	$16 \leq m \leq 31, n = 0$	INTEGER
BINARY(<i>m, n</i>)	$m = 0$ or $m > 32$ or $n \neq 0$	DECIMAL(<i>29, n</i>) ^{#5}

Note:

An error occurs at the target HiRDB if you attempt to import data that is longer than the target table column or whose numeric value overflows.

#1: If the RESTRUCT statement is specified with XDM/DS, Datareplicator uses the attribute and length defined with the RESTRUCT statement.

#2: Data conversion on the UNPACK type results in an error if any of the following conditions is satisfied:

- The value of the zoned part is not (F)₁₆.
- The value of the sign part is not (F)₁₆, (C)₁₆, or (D)₁₆.
- The value of the numeric part is outside the range (0)₁₆ to (9)₁₆.

If an error occurs, Datareplicator outputs an error message and the unimported information, and then terminates import processing on the erroneous import group. If you are using a UOC routine during import processing, Datareplicator passes the extracted data as is as UNPACK data to the UOC routine without converting the data type, and then resumes import processing.

#3: Spaces are not suppressed.

#4: The sign is not normalized.

#5: BINARY type is converted to DECIMAL type if the number of decimal places is not zero. Because the maximum precision for the DECIMAL type supported by XDM/DS is 29 decimal places, the digits beginning with the 30th decimal place are truncated.

#6: The maximum precision for the DECIMAL type supported by XDM/DS is 29.

(3) Correspondence of data types when the source database is XDM/RD E2

The following table shows the correspondence of data types between the source and target tables when the source database is XDM/RD E2.

Table 4-10: Correspondence of data types between the source and target tables when the source database is XDM/RD E2

Data type in source table	Data type in target table
INTEGER	INTEGER
SMALLINT	SMALLINT
DECIMAL	DECIMAL
LARGE DECIMAL	LARGE DECIMAL
FLOAT	FLOAT
SMALLFLT	SMALLFLT ^{#1}
CHAR	CHAR or MCHAR
VARCHAR	VARCHAR
LONG VARCHAR	
NCHAR	NCHAR
NVARCHAR	NVARCHAR
LONG NVARCHAR	
MCHAR	CHAR or MCHAR
MVARCHAR	VARCHAR or MVARCHAR
LONG MVARCHAR	
DATE	DATE
TIME	TIME
INTERVAL YEAR TO DAY	INTERVAL YEAR TO DAY

Data type in source table	Data type in target table
INTERVAL HOUR TO SECOND	INTERVAL HOUR TO SECOND
BLOB ^{#2}	BLOB ^{#2}

Notes:

- An error occurs at the target HiRDB if you attempt to import data that is longer than the target table column or whose numeric value overflows.
- You cannot specify repetition and array columns.

#1

Data linkage is supported, but rounding errors occur because values are rounded.

#2

To apply data linkage to columns of the BLOB type, specify ALL in the RECOVERY operand in the HiRDB table definition. For details about HiRDB table definition, see the manual *HiRDB Version 9 SQL Reference*.

(4) Correspondence of data types when the source database is ADM

The following table shows the correspondence of data types between the source and target tables when the source database is ADM.

Table 4-11: Correspondence of data types between the source and target tables when the source database is ADM

Data type under ADM		Data type at Datareplicator	Data type in target table
Cn		CHAR (n)	CHAR (n)
Xn		CHAR (n)	CHAR (n)
Pn	n ≤ 15	PACK (2n - 1, 0)	DECIMAL (2n - 1, 0)
	n > 15	CHAR (n)	CHAR (n)

Notes:

1. An error occurs at the target HiRDB if you attempt to import data that is longer than the target table column or whose numeric value overflows.
2. If you have used XDM/DS's RESTRUCT statement to define the UNPACK attribute as an ADM data type and the extracted update information contains data satisfying any of the following conditions, Datareplicator terminates import processing:
 - The value of the zoned part is not (F)₁₆.

- The value of the sign part is not (F)₁₆, (C)₁₆, or (D)₁₆.
- The value of the numeric part is outside the range (0)₁₆ to (9)₁₆.

If an error occurs, Datareplicator outputs an error message and the unimported information, and then terminates import processing on the erroneous import group. If you are using a UOC routine during import processing, Datareplicator passes the extracted data as is as UNPACK data to the UOC routine without converting the data type, and then resumes import processing.

(5) Correspondence of data types when the source database is PDM2 E2

Table 4-12: Correspondence of data types between the extraction redefinition fields and the target table's data types

Field attribute in extraction redefinition statement	Data length	Data type in target table
CHAR(<i>n</i>)	$n \leq 30,000$	CHAR(<i>n</i>)
NCHAR(<i>n</i>)	$n \leq 15,000$	NCHAR(<i>n</i>)
PACK(<i>m, n</i>)	N/A	DECIMAL(<i>m + n, n</i>)
PACKNS(<i>m, n</i>)	N/A	DECIMAL(<i>m + n, n</i>)
UNPACK(<i>m, n</i>)	N/A	DECIMAL(<i>m + n, n</i>)
UNPACKNS(<i>m, n</i>)	N/A	DECIMAL(<i>m + n, n</i>)
COMP(<i>m, n</i>)	$1 \leq m \leq 4, n = 0$	SMALLINT
	$5 \leq m \leq 9, n = 0$	INTEGER
	$m = 0$ or $m > 10$ or $n! = 0$	DECIMAL(29, <i>n</i>) [#]
BINARY(<i>m, n</i>)	$1 \leq m \leq 15, n = 0$	SMALLINT
	$16 \leq m \leq 31, n = 0$	INTEGER
	Other than the above	FLOAT
FLOAT(<i>n</i>)	$n = 4, 8$	FLOAT

Note:

An error occurs at the target HiRDB if you attempt to import data that is longer than the target table column or whose numeric value overflows.

#

The maximum precision for the DECIMAL type supported by XDM/DS is 29.

Table 4-13: Correspondence of data types between PDM2 E2's field attributes and the target table's data types

Field attribute in PDM2 E2	Data length	Data type in target table
$X(n)$	$n \leq 30,000$	CHAR(n)
$P(m, n), AP(m, n)$	$n > 0, 2m - 1 > n$	DECIMAL($2m - 1, n$)
	$n > 0, 2m - 1 \leq n$	DECIMAL(n, n)
	$n \leq 0$	DECIMAL($2m - 1 - n, 0$)
$Z(m, n), AZ(m, n)$	$n > 0, m > n$	DECIMAL(m, n)
	$n > 0, m \leq n$	DECIMAL(n, n)
	$n \leq 0$	DECIMAL($m - n, 0$)
$B(m, n), AB(m, n)$	$n > 0, 2.5m > n$	DECIMAL($2.5m, n$)
	$n > 0, 2.5m \leq n$	DECIMAL(n, n)
	$n \leq 0$	DECIMAL($2.5m - n, 0$)
$F(n)$	$n = 4, 8$	FLOAT
$N(n)$	$n \leq 30,000$	NCHAR($\uparrow n / 2 \uparrow$)

Note:

An error occurs at the target HiRDB if you attempt to import data that is longer than the target table column or whose numeric value overflows.

(6) Correspondence of data types when the source database is TMS-4V/SP

When the source database is TMS-4V/SP, you must use a UOC routine provided by TMS-4V/SP/Data Linkage Support to define data types. For details about data types when the source database is TMS-4V/SP, see the *VOS3 TMS-4V/SP/Data Linkage Support* manual.

(7) Correspondence of data types when the source database is RDB1 E2

The following table shows the correspondence of data types between the source and target tables when the source database is RDB1 E2.

Table 4-14: Correspondence of data types between the source and target tables when the source database is RDB1 E2

Data type in source table	Data type in target table
INTEGER	INTEGER

Data type in source table	Data type in target table
SMALLINT	SMALLINT
DECIMAL	DECIMAL
FLOAT	FLOAT
SMALLFLT	SMALLFLT [#]
CHAR	CHAR or MCHAR
VARCHAR	VARCHAR
LONG VARCHAR	N/A
NCHAR	NCHAR
NVARCHAR	NVARCHAR

N/A: Not subject to import processing.

Notes:

- An error occurs at the target HiRDB if you attempt to import data that is longer than the target table column or whose numeric value overflows.
- A repetition column cannot be specified.

#

Data linkage is supported, but rounding errors occur because values are rounded.

4.3.5 Designing for character code sets

This section explains how to design for the character code sets that are to be supported.

(1) *Types of character code sets*

The following types of character code sets can be used by the source and target databases:

- EBCDIC/KEIS, EBCDIK/KEIS

These character code sets are used by a source database that is a mainframe database. There are four available EBCDIC/KEIS and EBCDIK/KEIS character code sets:

- EBCDIC/KEIS78
- EBCDIK/KEIS78
- EBCDIC/KEIS83

- EBCDIK/KEIS83

These four character code sets are referred to collectively in this manual as the EBCDIK/KEIS character code set.

- EBCDIK

This character code set constitutes a mainframe's one-byte codes.

- JIS8/Shift JIS

These character code sets are used when the source or target database is HiRDB. They can be used when use of shift JIS Kanji encoding is specified in HiRDB's `pdsetup` command.

You must specify the appropriate HiRDB character code set with the `dblocale` operand in the extraction system definition for the source Datareplicator and with the `dblocale` operand in the import system definition for the target Datareplicator.

- EUC

This character encoding set is used when the source or target database is HiRDB. EUC is used when EUC Japanese Kanji encoding is specified in HiRDB's `pdsetup` command.

You must specify the appropriate HiRDB character code set with the `dblocale` operand in the extraction system definition for the source Datareplicator and with the `dblocale` operand in the import system definition for the target Datareplicator.

- UCS2

UCS2 is the Unicode UCS2 character set. Datareplicator supports only rules for encoding to UTF-8; it does not take into account mapping between codes and characters. The supported range of Unicode is UCS2 BMP (basic multi-language), and encoding of UCS2 to UTF-8 format is supported.

- UTF-8

This character string encoding is applicable when the source or target database is HiRDB and is used when UTF-8 is specified in HiRDB's `pdsetup` command. UTF-8 is one of the character encoding schemes that represent Unicode characters, and expresses a character as a combination of 8-bit information. The applicable HiRDB character encoding is specified in the `dblocale` operand of the extraction system definition for the source Datareplicator, and in the `dblocale` operand of the import system definition for the target Datareplicator.

(2) Character code conversion

If the source and target databases do not use the same character code set, Datareplicator converts character codes in accordance with a provided definition. If the source and

target databases use the same character code set, there is no need to convert character codes.

(a) Character code conversion by the target Datareplicator

The target Datareplicator executes character code conversion in accordance with the correspondence between the character code sets at the source and target databases. You use the `dblocale` operand in the import system definition to specify the target database's character code set.

If the source database is a mainframe database, use the `ebcdic_type` operand in the import environment definition to specify the type of EBCDIK/KEIS character code set for the source database. The following table shows the correspondence of character code sets between the source and target databases.

Table 4-15: Correspondence of character code sets between the source and target databases

Source database's character code set	Target database's character code set				
	EBCDIK/KEIS ^{#1}	EBCDIK	JIS8/Shift JIS	EUC ^{#2}	UTF-8
EBCDIK/KEIS ^{#1}	--	--	Y	Y	Y
EBCDIK	--	--	Y	--	--
JIS8/Shift JIS	Y	Y	--	Y	Y
EUC ^{#2}	Y	--	Y	--	Y
UTF-8	Y	--	Y	Y	--

Y: The character code set used in the source database can be converted to the character code set used in the target database.

--: There is no need to convert the character code set.

#1

Only XDM/DS and VOS3 Database Datareplicator support EBCDIK/KEIS.

#2

Conversion results might not be correct if the specification of the character code set is not in the range 0 to 2.

(3) Rules for character code conversion

Datareplicator uses the following character code conversion rules:

Character code conversion method

Uses a mapping table for converting character codes.

Definition of Gaiji conversion method

For conversion of Gaiji characters, you must use the `hdscconvdt` command to edit and define a mapping table for converting character codes.

(4) Rules for character code conversion from EBCDIK/KEIS to JIS8/Shift JIS

(a) Single-byte codes

Datareplicator converts a single-byte code to the corresponding Shift JIS character code.

(b) Double-byte codes (standard character codes)

- Datareplicator converts a double-byte code to the corresponding Shift JIS character code.
- Function characters $((0A42)_{16}$ and $(0A41)_{16}$) are deleted and shifted up. If this results in a remainder, it is converted to a space character $((20)_{16})$.
- Codes $(00)_{16}$ to $(40)_{16}$ enclosed by function characters are converted to the corresponding Shift JIS character codes.
- If the last character is the first byte of a double-byte code, Datareplicator converts it to a space character $((20)_{16})$.

(c) Double-byte codes (Gaiji)

Datareplicator uses the provided mapping table for converting character codes to convert double-byte Gaiji codes. If a Gaiji character is not defined or no mapping table for converting character codes has been provided, Datareplicator converts the Gaiji code to a space character according to the value of the `undefcode_cnv` operand specified in the import environment definition.

(d) Space character

Datareplicator converts the space character in shift code. The following table shows the conversion rules for the space character (from EBCDIK/KEIS to JIS8/Shift JIS).

Table 4-16: Conversion rules for the space character (from EBCDIK/KEIS to JIS8/Shift JIS)

EBCDIK/KEIS character code	JIS8/Shift JIS character code
Double-byte space character $(A1A1)_{16}$	Double-byte space character $(8140)_{16}$

EBCDIK/KEIS character code	JIS8/Shift JIS character code
Two consecutive single-byte space characters (40) ₁₆ (40) ₁₆	Converted to one double-byte space character or two consecutive single-byte space characters, depending on the <code>shiftspace_cnv</code> operand value specified in the import environment definition
Single-byte space character (40) ₁₆	Single-byte space character (20) ₁₆

(5) Rules for character code conversion from EBCDIK/KEIS to EUC

(a) Single-byte codes

- Datareplicator converts a single-byte code, excluding kana characters, to the corresponding EUC character code.
- Datareplicator converts a kana character to a double-byte code.

(b) Double-byte codes (standard character codes)

- Datareplicator converts a double-byte code to the corresponding EUC character code.
- Function characters ((0A42)₁₆ and (0A41)₁₆) are deleted and shifted up. If this results in a remainder, it is converted to a space character ((20)₁₆).
- Codes (00)₁₆ to (40)₁₆ enclosed by function characters are converted to the corresponding Shift JIS character codes.
- If the last character is the first byte of a double-byte code, Datareplicator converts it to a space character ((20)₁₆).

(c) Double-byte codes (Gaiji)

Datareplicator assigns the last 8,836 characters in the 9,024-character Gaiji area as the Gaiji area for EUC character codes for conversion purposes.

Datareplicator converts any other code in accordance with the user-created mapping table for converting character codes. If a Gaiji character is not defined or no mapping table for converting character codes has been provided, Datareplicator converts the Gaiji code to a space character according to the value of the `undefcode_cnv` operand specified in the import environment definition.

Note that code set 3 might result in an error when an SQL statement is issued.

(d) Space character

Datareplicator converts the space character. The following table shows the conversion rules for the space character (from EBCDIK/KEIS to EUC).

Table 4-17: Conversion rules for the space character (from EBCDIK/KEIS to EUC)

EBCDIK/KEIS character code	JIS8/Shift JIS character code
Double-byte space character (A1A1) ₁₆	Double-byte space character (A1A1) ₁₆
Two consecutive single-byte space characters (40) ₁₆ (40) ₁₆	Converted to one double-byte space character or two consecutive single-byte space characters, depending on the <code>shiftspace_cnv</code> operand value specified in the import environment definition
Single-byte space character (40) ₁₆	Single-byte space character (20) ₁₆

(e) Handling of overflow

Datereplicator converts a single-byte kana character to a double-byte character. If the source table contains single-byte kana characters, the data length increases after conversion and data overflow might occur. The following table shows how overflow is handled (from EBCDIK/KEIS to EUC).

Table 4-18: Handling of overflow (from EBCDIK/KEIS to EUC)

Location of overflow	Datereplicator's action	User's action
Literal or update data	SQL error (applicable if the data exceeds the defined length after conversion of all bytes)	Change the defined length and re-execute import processing.

Note:

If an identifier exceeds 30 bytes or update data exceeds 32,000 bytes, Datereplicator is unable to continue processing. Before starting data linkage, make sure that source system identifiers and update data will not exceed the permitted maximum length.

(6) Rules for character encoding conversion from EBCDIK/KEIS to UTF-8

(a) Single-byte codes

- Datereplicator converts single-byte codes, excluding kana characters, to the corresponding UTF-8 character codes.
- Datereplicator converts a kana character to a three-byte code.

(b) Double-byte codes (standard character codes)

- Datereplicator converts double-byte codes to the corresponding UTF-8 character codes.

- Function characters $((0A42)_{16}$ and $(0A41)_{16}$) are deleted and shifted up. If this results in a remainder, the remainder is converted to a space character $((20)_{16})$.
- Codes $(00)_{16}$ to $(40)_{16}$ enclosed by function characters are converted to the corresponding UTF-8 character codes.
- If the last character is the first byte of a double-byte code, Datareplicator converts it to a space character $((20)_{16})$.

(c) Double-byte codes (Gaiji)

Datareplicator uses the provided mapping table for converting character codes to convert double-byte Gaiji codes.

If a Gaiji character is not defined or no mapping table for converting character codes has been provided, Datareplicator converts the Gaiji code to a space character according to the value of the `undefcode_cnv` operand specified in the import environment definition.

(d) Space character

Datareplicator converts the space character in shift code. The following table shows the conversion rules for the space character (from EBCDIK/KEIS to UTF-8).

Table 4-19: Conversion rules for the space character (from EBCDIK/KEIS to UTF-8)

EBCDIK/KEIS character code	UTF-8 character code
Double-byte space character $(A1A1)_{16}$	Double-byte space character $(E38080)_{16}$
Two consecutive single-byte space characters $(40)_{16}(40)_{16}$	Converted to one double-byte space character or two consecutive single-byte space characters, depending on the <code>shiftspace_cnv</code> operand value specified in the import environment definition
Single-byte space character $(40)_{16}$	Single-byte space character $(20)_{16}$

(e) Handling of overflow

Datareplicator converts a single-byte kana character to a three-byte character and a double-byte standard kanji character to a three-byte character. If the source table contains single-byte kana characters or standard kanji characters, the data length increases after conversion and data overflow might occur.

The following table shows how overflow is handled (from EBCDIK/KEIS to UTF-8).

Table 4-20: Handling of overflow (from EBCDIK/KEIS to UTF-8)

Location of overflow	Datereplicator's action	User's action
Literal or update data	SQL error (applicable if the data exceeds the defined length after conversion of all bytes)	Change the defined length and re-execute import processing.

Note:

If an identifier exceeds 30 bytes or update data exceeds 32,000 bytes, Datereplicator is unable to continue processing. Before starting data linkage, make sure that source system identifiers and update data will not exceed the permitted maximum lengths.

(f) Notes

If the target database is HiRDB using UTF-8, a column of the NCHAR or NVARCHAR type cannot be created in the table. To store such data in a different data type, such as MCHAR, use a column data editing UOC routine.

(7) Rules for character encoding conversion from EBCDIK to JIS8**(a) Single-byte codes**

Datereplicator converts such single-byte codes to the corresponding JIS8 character codes (single-byte codes). Because only single-byte codes can be specified in a column for which EBCDIK is specified in the character set specification, the following might result:

- All characters are treated as being single-byte codes. If double-byte codes are converted, garbled characters might result.
- In single-byte code conversion, the length of the data remains the same before and after the conversion.
- No character code conversion error occurs because all characters are processed as being single-byte codes.

(8) Rules for character encoding conversion from JIS8 to EBCDIK**(a) Single-byte codes**

Datereplicator converts single-byte codes to the corresponding EBCDIK character codes.

(b) Double-byte codes (standard character codes)

Datereplicator treats bytes 1 and 2 of a double-byte code as single-byte codes and converts them to single-byte EBCDIK code characters. Therefore, the following might result:

- Converting double-byte codes results in garbled characters.

- In single-byte code conversion, the length of the data remains the same before and after the conversion.
- No character code conversion error occurs because all characters are processed as being single-byte codes.

(9) Rules for character code conversion from JIS8/Shift JIS or EUC to EBCDIK/KEIS

(a) Single-byte codes

Datareplicator converts a single-byte code to the corresponding EBCDIK/KEIS character code.

(b) Double-byte codes (standard character codes)

Datareplicator converts a double-byte code to the corresponding EBCDIK/KEIS character code. If the last character is the first byte of a double-byte code, Datareplicator converts it to a space character ((40)₁₆).

(c) Double-byte codes (Gaiji)

Datareplicator converts a double-byte Gaiji code in accordance with the mapping table for converting character codes. If a Gaiji character is not defined or no mapping table for converting character codes has been provided, Datareplicator converts the Gaiji code to the space character ((4040)₁₆).

(d) Space character

Datareplicator converts a double-byte space character ((8140)₁₆) to the corresponding double-byte space character ((A1A1)₁₆) and a single-byte space character ((20)₁₆) to the corresponding single-byte space character ((40)₁₆).

(10) Rules for character encoding conversion from JIS8/Shift JIS to UTF-8

(a) Single-byte codes

- Datareplicator converts each single-byte code, excluding kana characters, to the corresponding UTF-8 character code.
- Datareplicator converts a kana character to a three-byte code.

(b) Double-byte codes (standard character codes)

Datareplicator converts double-byte codes to the corresponding UTF-8 character codes.

(c) Double-byte codes (Gaiji)

Datareplicator uses the provided mapping table for converting character codes to convert double-byte Gaiji codes.

If a Gaiji character is not defined or no mapping table for converting character codes

has been provided, Datareplicator converts the Gaiji code to a space character according to the value of the `undefcode_cnv` operand specified in the import environment definition.

(d) Space character

Datareplicator converts the space character in shift code. The following table shows the conversion rules for the space character (from JIS8/Shift JIS to UTF-8).

Table 4-21: Conversion rules for the space character (from JIS8/Shift JIS to UTF-8)

JIS8/SJIS character code	UTF-8 character code
Double-byte space character (8140) ₁₆	Double-byte space character (E38080) ₁₆
Single-byte space character (20) ₁₆	Single-byte space character (20) ₁₆

(e) Handling of overflow

Datareplicator converts a single-byte kana character to a three-byte character and a double-byte standard kanji character to a three-byte character. If the source table contains single-byte kana characters or standard kanji characters, the data length increases after conversion and data overflow might occur. The following table shows how overflow is handled (from JIS8/Shift JIS to UTF-8).

Table 4-22: Handling of overflow (from JIS8/Shift JIS to UTF-8)

Location of overflow	Datareplicator's action	User's action
Literal or update data	SQL error (applicable if the data exceeds the defined length after conversion of all bytes)	Change the defined length and re-execute import processing.

Note:

If an identifier exceeds 30 bytes or update data exceeds 32,000 bytes, Datareplicator is unable to continue processing. Before starting data linkage, make sure that source system identifiers and update data will not exceed the permitted maximum lengths.

(f) Notes

- If the target database is HiRDB using UTF-8, a column of the NCHAR or NVARCHAR type cannot be created in the table. To store such data in a different data type, such as MCHAR, use a column data editing UOC routine.
- There are two types of mapping from SJIS Kanji encoding to Unicode (UCS2), as

shown in the following table.

Table 4-23: Character mapping in SJIS Kanji encoding and Unicode

Mapping method			Description
Method	Target	Range of kanji	
JIS method	SJIS to JIS X0221	JIS standard level 1	Depends on the mapping stipulated by JIS X0221.
		JIS standard level 2	
MS method	Windows-the customer support center character set to MS-The customer support center	JIS standard level 1	Depends on the mapping defined by Microsoft.
		JIS standard level 2	
		Vendor-extended characters	

In the MS method, vendor-extended characters have been added; when characters are converted to UCS2, the code point that is set is not the same as with the JIS method. Datareplicator uses the MS method.

- If data of the NCHAR type is converted and the length of UTF-8 data obtained after conversion consists of an odd number of bytes, a single-byte space is added to the resulting data. What is set in the `colLen` member (column data length) is the *length of the data obtained after conversion (in bytes) + 1 ÷ 2*.

If the length of UTF-8 data obtained after conversion consists of an even number of bytes, the resulting data is used as is. What is set in the `colLen` member (column data length) is the *length of the data obtained after conversion (in bytes) ÷ 2*.

- If the definition length n of the NCHAR type in the source table is an even number, the definition length of the MCHAR type in the target table becomes $n \times 3$.

If the definition length n of the NCHAR type in the source table is an odd number, the definition length of the MCHAR type in the target table becomes $(n \times 3) + 1$.

(11) Rules for character code conversion from EUC to JIS8/Shift JIS

(a) Single-byte codes

Datareplicator converts a single-byte code to the corresponding Shift JIS character code.

(b) Double-byte codes (standard character codes)

Datareplicator converts a double-byte code to the corresponding Shift JIS character code. If the last character is the first byte of a double-byte code, Datareplicator converts it to a space character $((20)_{16})$.

(c) Double-byte codes (Gaiji)

Datareplicator converts a double-byte Gaiji code in accordance with the mapping table for converting character codes. If a Gaiji character is not defined or no mapping table for converting character codes has been provided, Datareplicator converts the Gaiji code to a space character according to the value of the `undefcode_cnv` operand specified in the import environment definition.

(d) Space character

Datareplicator converts a double-byte space character $((A1A1)_{16})$ to the corresponding code $((8140)_{16})$. Datareplicator converts two consecutive single-byte space characters $((40)_{16})$ to a double-byte space character $((8140)_{16})$. One single-byte space character $((40)_{16})$ is converted to the corresponding code $((20)_{16})$.

(12) Rules for character code conversion from EUC to UTF-8**(a) Single-byte codes**

Datareplicator converts a single-byte code, excluding kana characters, to the corresponding UTF-8 character code.

(b) Double-byte codes (standard character codes)

- Datareplicator converts a kana character to a three-byte code.
- Datareplicator converts a double-byte code to the corresponding UTF-8 character code.

(c) Three-byte codes (Gaiji)

Datareplicator converts a double-byte code (Gaiji) in accordance with the user-created mapping table for converting character codes.

If a Gaiji character is not defined or no mapping table for converting character codes has been provided, Datareplicator converts the Gaiji code to a space character according to the value of the `undefcode_cnv` operand specified in the import environment definition.

(d) Space character

Datareplicator converts the space character in shift code. The following table shows the conversion rules for the space character (from EUC to UTF-8).

Table 4-24: Conversion rules for the space character (from EUC to UTF-8)

EUC character code	UTF-8 character code
Double-byte space character (A1A1) ₁₆	Double-byte space character (E38080) ₁₆
Single-byte space character (20) ₁₆	Single-byte space character (20) ₁₆

(e) Handling of overflow

Datareplicator converts a double-byte kana character to a three-byte character and a double-byte standard kanji character to a 3-byte character. If the source table contains single-byte kana characters or standard kanji characters, the data length increases after conversion and data overflow might occur. The following table shows how overflow is handled (from EUC to UTF-8).

Table 4-25: Handling of overflow (for conversion from EUC to UTF-8)

Location of overflow	Datareplicator's action	User's action
Literal or update data	SQL error (applicable if the data exceeds the defined length after conversion of all bytes)	Change the defined length and re-execute import processing.

Note:

If an identifier exceeds 30 bytes or update data exceeds 32,000 bytes, Datareplicator is unable to continue processing. Before starting data linkage, make sure that source system identifiers and update data will not exceed the permitted maximum lengths.

(f) Notes

- If the target database is HiRDB using UTF-8, a column of the NCHAR or NVARCHAR type cannot be created in the table. To store data in a different data type, such as MCHAR, use a column data editing UOC routine.
- If data of the NCHAR type is converted and the length of UTF-8 data obtained after conversion consists of an odd number of bytes, a single-byte space is added to the resulting data. What is set in the `col_len` member (column data length) is the *length of the data obtained after conversion (in bytes) + 1 ÷ 2*.
If the length of UTF-8 data obtained after conversion consists of an even number of bytes, the resulting data is used as is. What is set in the `col_len` member (column data length) is the *length of the data obtained after conversion (in bytes) ÷ 2*.
- If the definition length *n* of the NCHAR type in the source table is an even number,

the definition length of the MCHAR type in the target table becomes $n \times 3$.

If the definition length n of the NCHAR type in the source table is an odd number, the definition length of the MCHAR type in the target table becomes $(n \times 3) + 1$.

(13) Rules for character code conversion from UTF-8 to EUC or JIS8/Shift JIS

(a) Single-byte codes

Datareplicator converts a single-byte code to the corresponding character code.

(b) Double-byte and three-byte codes (standard kanji)

Datareplicator converts a double-byte or three-byte code to the corresponding UTF-8 character code.

(c) Three-byte codes (Gaiji)

Datareplicator converts a three-byte code in accordance with the user-created mapping table for converting character codes.

If a Gaiji character is not defined or no mapping table for converting character codes has been provided, Datareplicator converts the Gaiji code to a space character according to the value of the `undefcode_cnv` operand specified in the import environment definition.

(14) Suppression of character code conversion

You can suppress character code conversion for a column at the target Datareplicator by specifying suppression of character code conversion for the desired extracted column. For details about the specification to suppress character code conversion, see the update information field definition in *5.10 Import definition*.

(15) Details of conversion rules for each character the customer support center

(a) Conversion rules for JIS8/Shift JIS codes

The following table shows the conversion rules for JIS8/Shift JIS codes.

Table 4-26: Conversion rules for JIS8/Shift JIS codes

1-byte	2-byte	3-byte	Code conversion rule
0x00 to 0x80	--	--	Treats as JIS8 and converts it to the corresponding code.
0x81 to 0x9F	0x40 to 0xFC (excluding 0x7F)	--	Treats as SJIS (Kanji) and converts it to the corresponding code.
	Other	--	Converts according to the specification of <code>undefcode_cnv</code> .
	--	--	Treats as incomplete code and skips it without converting.

1-byte	2-byte	3-byte	Code conversion rule
0xA0 to 0xDF	--	--	Treats as JIS8 and converts it to the corresponding code.
0xE0 to 0xEF	0x40 to 0xFC (excluding 0x7F)	--	Treats as SJIS (Kanji) and converts it to the corresponding code.
	Other	--	Converts according to the specification of <code>undefcode_cnv</code> .
	--	--	Treats as incomplete code and skips it without converting.
0xF0 to 0xFC	0x40 to 0xFC (excluding 0x7F)	--	Treats as SJIS (Gaiji) and converts it to the corresponding code.
	Other	--	Converts according to the specification of <code>undefcode_cnv</code> .
	--	--	Treats as incomplete code and skips it without converting.
0xFD to 0xFF	--	--	Treats as JIS8 and converts it to the corresponding code.

(b) Conversion rules for EUC codes

The following table shows the conversion rules for EUC codes.

Table 4-27: Conversion rules for EUC codes

1-byte	2-byte	3-byte	Code conversion rule
0x00 to 0x8D	--	--	Treats as code set 0 and converts it to the corresponding code.
0x8E	0xA0 to 0xFF	--	Treats as code set 2 (kana characters) and converts it to the corresponding code.
	Other	--	Converts according to the specification of <code>undefcode_cnv</code> .
	--	--	Treats as incomplete code and skips it without converting.
0x8F	0xA1 to 0xFE	0xA1 to 0xFE	Treats as code set 3 (Gaiji) and converts it to the corresponding code.
		Other	Converts according to the specification of <code>undefcode_cnv</code> .

1-byte	2-byte	3-byte	Code conversion rule
		--	Treats as incomplete code and skips it without converting.
	Other	0xA1 to 0xFE	Converts according to the specification of <code>undefcode_cnv</code> .
		Other	
		--	Treats as incomplete code and skips it without converting.
--	--		
0x90 to 0x9F	--	--	Treats as code set 0 and converts it to the corresponding code.
0xA0	--	--	Converts to 0x20.
0xA1 to 0xFE	0xA1 to 0xFE	--	Treats as code set 1 (Kanji) and converts it to the corresponding code.
	Other	--	Converts according to the specification of <code>undefcode_cnv</code> .
	--	--	Treats as incomplete code and skips it without converting.

(c) Conversion rules for EBCDIK/KEIS codes

The following table shows the conversion rules for EBCDIK/KEIS codes.

Table 4-28: Conversion rules for EBCDIK/KEIS codes

1-byte	2-byte	3-byte	Code conversion rule	
Single-byte shift on	0x00-0x09	--	Converts to the corresponding code.	
	0x0A	0x41	--	Toggles single-byte shift without performing conversion.
		0x42	--	Toggles double-byte shift without performing conversion.
		Other	--	Converts byte 1 or byte 2 as a single-byte code.
		--	--	Treats as incomplete code and skips it without converting.

1-byte		2-byte	3-byte	Code conversion rule
	0x0B to 0xFF	--	--	Converts to the corresponding code.
Double-byte shift on	0x00 to 0x40	--	--	Converts byte 1 or byte 2 as a single-byte code.
	0x41 to 0xA0	0xA1 to 0xFE	--	Converts to the corresponding code.
		Other	--	Converts according to the specification of <code>undefcode_cnv</code> .
		--	--	Treats as incomplete code and skips it without converting.
	0xA1 to 0xFE	0xA1 to 0xFE	--	Converts to the corresponding code.
		Other	--	Converts according to the specification of <code>undefcode_cnv</code> .
		--	--	Treats as incomplete code and skips it without converting.
	0xFF	0x01 to 0xFF	--	Converts according to the specification of <code>undefcode_cnv</code> .
		--	--	Treats as incomplete code and skips it without converting.

(d) Conversion rules for UTF-8 codes

The following table shows the conversion rules for UTF-8 codes.

Table 4-29: Conversion rules for UTF-8 codes

1-byte	2-byte	3-byte	Code conversion rule
0x00 to 0x7F	--	--	Treats as single-byte code and converts it to the corresponding code.
0x80 to 0xBF	0x00 to 0xFF	--	Converts to 0x20.
	--	--	

1-byte	2-byte	3-byte	Code conversion rule
0xC2 to 0xDE	0x80 to 0xFF	--	Treats as double-byte code and converts it to the corresponding code.
	Other	--	Converts according to the specification of <code>undefcode_cnv</code> .
	--	--	Treats as incomplete code and skips it without converting.
0xDF	0x80 to 0xBF	--	Treats as double-byte code and converts it to the corresponding code.
	Other	--	Converts according to the specification of <code>undefcode_cnv</code> .
	--	--	Treats as incomplete code and skips it without converting.
0xE0	0xA0 to 0xFF	0x80 to 0xFF	Treats as three-byte code and converts it to the corresponding code.
		Other	Converts according to the specification of <code>undefcode_cnv</code> .
		--	Treats as incomplete code and skips it without converting.
	Other	0x80 to 0xFF	Converts according to the specification of <code>undefcode_cnv</code> .
		Other	
		--	Treats as incomplete code and skips it without converting.
	--	--	--
0xE1 to 0xEE	0x80 to 0xFF	0x80 to 0xFF	Treats as three-byte code and converts it to the corresponding code.
		Other	Converts according to the specification of <code>undefcode_cnv</code> .
		--	Treats as incomplete code and skips it without converting.
	Other	0x80 to 0xFF	Converts according to the specification of <code>undefcode_cnv</code> .
		--	Treats as incomplete code and skips it without converting.
	--	--	--

1-byte	2-byte	3-byte	Code conversion rule
0xEF	0x80 to 0xBF	0x80 to 0xBF	Treats as three-byte code and converts it to the corresponding code.
		Other	Converts according to the specification of <code>undefcode_cnv</code> .
		--	Treats as incomplete code and skips it without converting.
	Other	0x80 to 0xBF	Converts according to the specification of <code>undefcode_cnv</code> .
		Other	
		--	Treats as incomplete code and skips it without converting.
--	--	--	
Other	--	--	Converts according to the specification of <code>undefcode_cnv</code> .

4.3.6 Designing repetition columns

This subsection explains how to design repetition columns.

(1) Limitations on extracting repetition column data

Datareplicator does not require any special definitions for extraction of repetition columns. However, the following limitations apply:

- A repetition column that is to be extracted must be handled as a single entity. The individual elements that make up the repetition column cannot be extracted.
- A repetition column is not subject to the mapping key.

(2) Limitations on importing repetition column data

When you import repetition column data that has been updated by the `UPDATE` statement's `SET` clause and the following conditions are true, an SQL error (`SQLCODE = -129`) occurs at the target Datareplicator, which disables import processing:

- (Number of mapping key columns + number of columns to be extracted from the table subject to extraction other than repetition columns + number of update items corresponding to the repetition column specified in the `UPDATE` statement's `SET` clause) > 30,000

Because the number of mapping keys is always at least one, the maximum number of update items in a repetition column in the `SET` clause that can be imported is 29,999. To update a large amount of element data with a single SQL statement, such as when updating all elements in a repetition column in the source system, we recommend that

you update column-by-column instead of by specifying an update item for each element.

(3) Import definition for linking repetition columns

To specify a repetition column in the import definition, use the `load` statement to define that the table that contains the repetition column is to be subject to import processing, in the same manner as with the conventional method. You can specify only the `NULL` literal for a repetition column in the update information field definition (`const` specified in the `format` statement's `name` clause). If you specify any other value, Datareplicator imports the specified literal as is without regarding it as an error (this results in a data compatibility error at the target HiRDB).

If you specify a table that contains a repetition column with the `load` statement in the import definition, `timestamp` cannot be specified to create a time-ordered table. If you do specify `timestamp` for a table containing a repetition column, Datareplicator regards it as a definition analysis error and terminates import processing with the corresponding data linkage identifier (`dsid`).

When you specify a table that contains a repetition column with the `load` statement in the import definition, you can still create a merge table by specifying `sqlconvopt1` or `sqlconvopt2`. However, to change `UPDATE` to `INSERT` in an element specification, the null value will be provided for any absent element data (see (3) in 3.3.11 *Creating a merge table*).

(4) Character encoding conversion for data in repetition columns

If the source HiRDB uses a different character encoding set than the target HiRDB, Datareplicator performs character encoding conversion on each element data item in a repetition column. For details about character encoding conversion, see 4.3.5 *Designing for character code sets*.

(5) UOC interface for repetition column data

You can use an import information editing UOC routine and a column data editing UOC routine to process repetition column data. However, the column data editing UOC routine passes each element data item individually (each call to the column data editing UOC function passes one element data item to this UOC function). This means that you cannot use a column data editing UOC routine to increase or decrease the number of elements.

(6) Processing when the numbers of elements do not match

If you execute `UPDATE SET` or `DELETE` specifying an element number with no data for a repetition column, HiRDB ignores update processing on any nonexistent repetition column element (in which case 0 is set in `SQLCODE` and `W` in `SQLWARN7`).

With normal data linkage, the numbers of elements in a repetition column always match between the source and target. However, if a mismatch occurs due to importing of a merge table or updating of the target table conducted by HiRDB, Datareplicator

executes processing according to the `skip_mvcelmwarn` and `sqlerr_skip_info` operand specifications in the import environment definition.

skip_mvcelmwarn		sqlerr_skip_info			
		output	msgoutput	sqloutput	nooutput
true	Processing	Normal (resume import processing)			
	Output message	Warning	Warning	None	None
	Unimported output	Output	None	Output	None
false	Processing	Error (roll back the transaction and cancel import processing)			
	Output message	Error			
	Unimported output	Output			

4.3.7 Creating a time-ordered information table

This section explains the procedure for creating a time-ordered information table.

(1) Available time-ordered information

By specifying `timestamp` in the `to` clause of the import table definition, you can obtain a historical record of update information in chronological order. However, time-ordered information might not be available, depending on the type of update processing. If Datareplicator cannot obtain the specified time-ordered information, it sets the null value.

The following table shows the contents and availability of time-ordered information.

Table 4-30: Contents and availability of time-ordered information

Information	Description	Availability			
		Update	Insert	Delete	PURGE
Extraction date	Date update information was output to the update journal at the source system	Y	Y	Y	Y
Extraction time	Time update information was output to the update journal at the source system	Y	Y	Y	Y
Extraction time stamp	Date and time update information was output to the update journal at the source system	Y	Y	Y	Y
Import date	Date update information was imported into the target database at the target system	Y	Y	Y	Y

Information	Description	Availability			
		Update	Insert	Delete	PURGE
Import time	Time update information was imported into the target database at the target system	Y	Y	Y	Y
Import time stamp	Date and time update information was imported to the target database at the target system	Y	Y	Y	Y
Import type	Import type, indicated as follows: Update: upd Insert: ins Delete: del PURGE: purge	Y	Y	Y	Y
Mapping key	Key value used to identify the data to be updated	Y	Y	Y	N/A
Update information	Updated data	Y	Y	N/A	N/A

Y: Available.

N/A: Becomes the null value.

(2) Defining a table for collection of time-ordered information

To collect time-ordered information, such as update information, import dates, and import times, you must create in advance in the target HiRDB database a table for storing this information (time-ordered information table). You can specify any name for the table and you must specify the table's column structure (number of columns, order of columns, column names). You must define the column attributes as appropriate for the information to be collected. Depending on the type of information, you might have to specify a literal in the `const` clause in the update information field definition.

The following table shows the column attributes of a time-ordered information table and the specification in the `const` clause.

Table 4-31: Column attributes for a time-ordered information table and the specification in the `const` clause

Information to be collected	Column attribute	Specification in the <code>const</code> clause in the update information field definition
Extraction date	DATE	<code>extract_date</code>
Extraction time	TIME	<code>extract_time</code>

Information to be collected	Column attribute	Specification in the const clause in the update information field definition
Extraction time stamp	TIMESTAMP[(p)] [#] , $p = 0, 2, 4, 6$	extract_timestamp
Import date	DATE	reflect_date
Import time	TIME	reflect_time
Import time stamp	TIMESTAMP[(p)] [#] , $p = 0, 2, 4, 6$	reflect_timestamp
Import type	CHAR(5)	reflect_kind
Mapping key	Attribute that allows storage of mapping keys	N/A
Update information	Attribute that allows storage of update information	N/A

N/A: Not applicable.

#

p is an integer representing the number of decimal places displayed for the seconds value. For example, $p = 2$ means that the seconds value is to be obtained out to two decimal places.

Note the following about using time-ordered information:

- You use HiRDB's retrieval facility (`SELECT` command) to reference time-ordered information. However, the information might not be collected in the order in which the source database was updated. To retrieve the information in the source database's updating order, you must sort it during retrieval.
- Information, such as update information, import dates, and import times, keeps accumulating over time in a time-ordered information table. Therefore, you need to periodically delete unneeded information from this table by issuing the `DELETE` or `PURGE TABLE` statement.

(3) Notes about the data that is imported to a time-ordered information table

- If data obtained from BLOB column partitions has been imported to a time-ordered table, only the data obtained from the BLOB column partitions can be imported.
- Do not import to a time-ordered information table BLOB-type or BINARY-type data to which backward deletion updating has been performed. If such data is imported to a time-ordered information table, inconsistency occurs between the source and target databases.

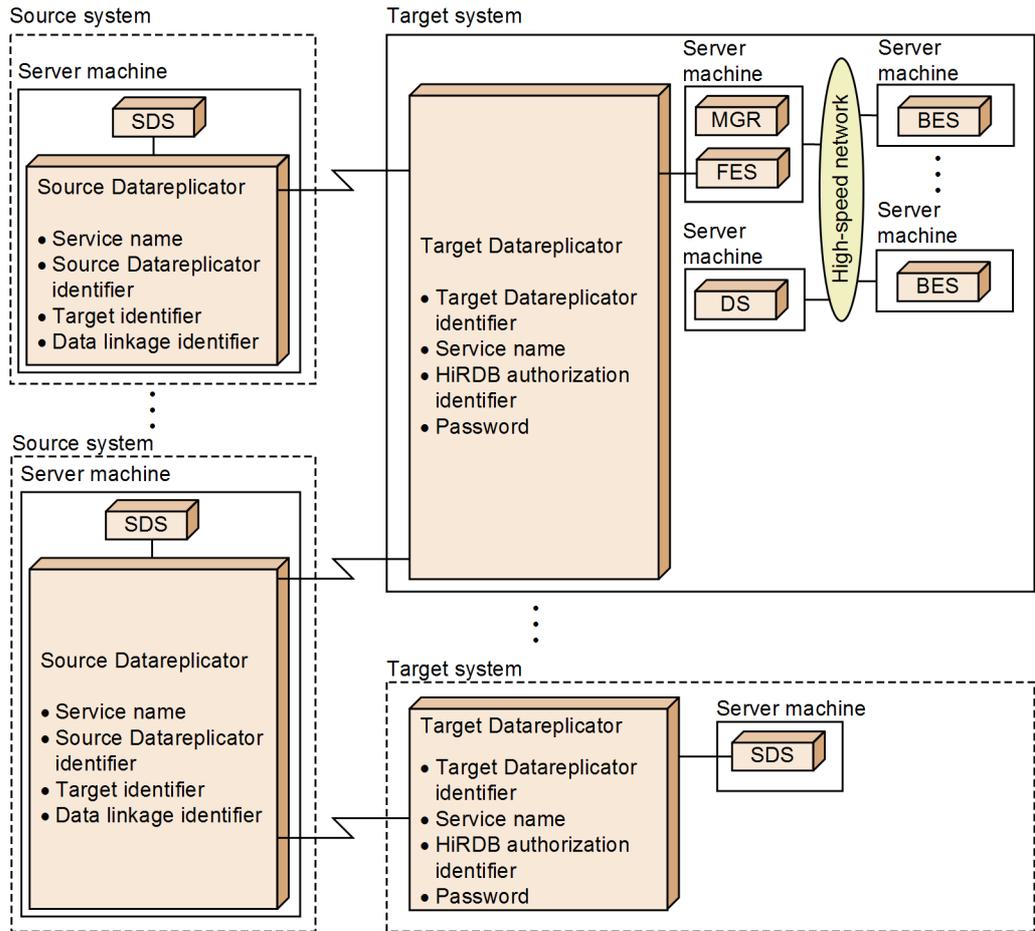
4.4 Designing the correspondence between source and target systems

This section explains the procedure for designing the correspondence between the source and target systems.

4.4.1 Designing data linkage from one HiRDB to another HiRDB

In designing data linkage from one HiRDB to another HiRDB, the correspondence between the source and target systems depends on whether the source HiRDB is a single server or a parallel server, as shown in Figures 4-15 and 4-16, respectively.

Figure 4-15: Correspondence between the source and target systems in designing data linkage from one HiRDB to another HiRDB: Source HiRDB is a single server

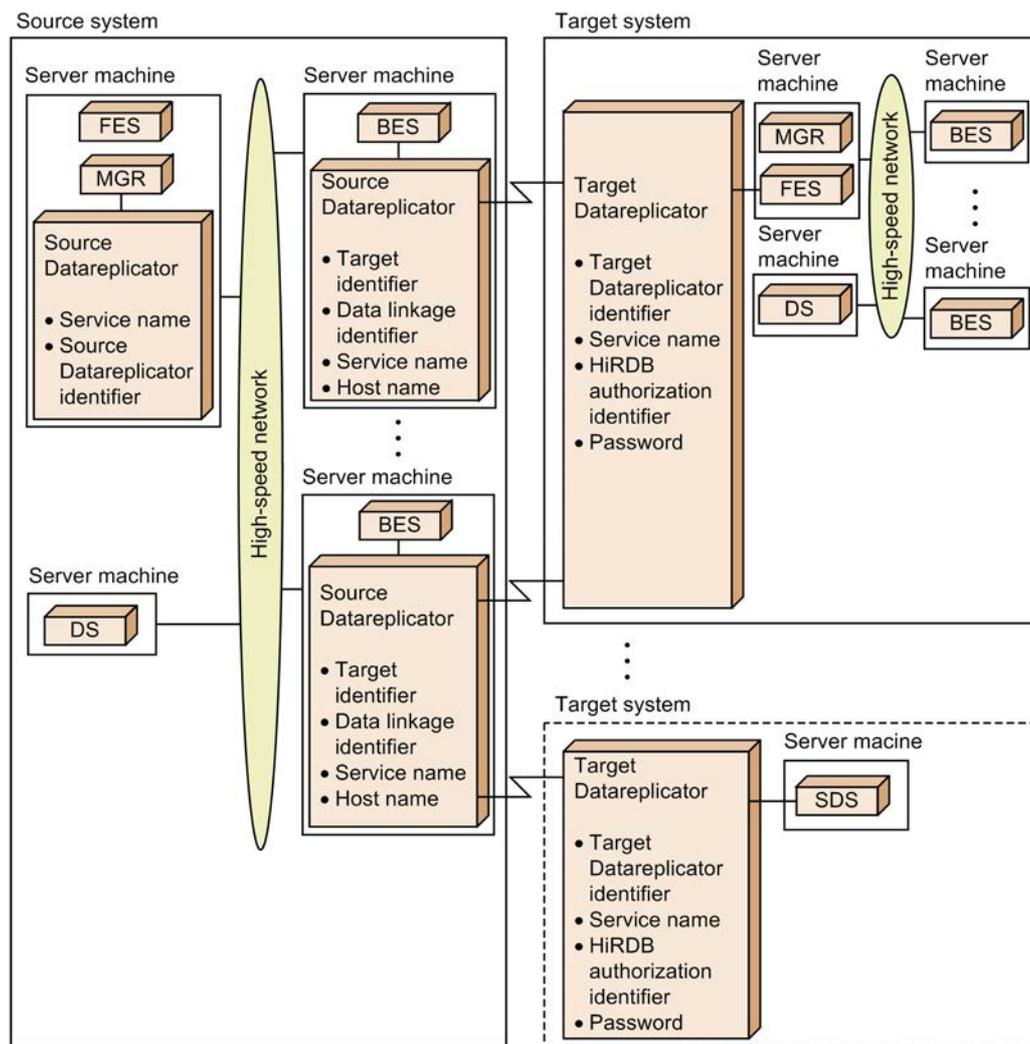


Legend:

SDS : HiRDB single server
 FES : HiRDB front-end server
 DS : HiRDB dictionary server

MGR : HiRDB system manager
 BES : HiRDB back-end server

Figure 4-16: Correspondence between the source and target systems in designing data linkage from one HiRDB to another HiRDB: Source HiRDB is a parallel server



Legend:
 SDS : HiRDB single server MGR : HiRDB system manager
 FES : HiRDB front-end server BES : HiRDB back-end server
 DS : HiRDB dictionary server

(1) Correspondence between source HiRDB and source Datareplicator

This subsection explains the correspondence between the source HiRDB and the

source Datareplicator.

(a) Proportion when source HiRDB is a single server

Source HiRDB (single server) to source Datareplicator = 1 to 1

There can be only one source Datareplicator per source HiRDB. There can be only one source HiRDB per source Datareplicator.

(b) Proportion when source HiRDB is a parallel server

Source HiRDB (parallel server) to source Datareplicators = 1 to $n^{\#}$

$\#$: $n = 1 +$ number of servers containing a back-end server

For one source HiRDB, one source Datareplicator must operate at each server that contains the system manager or a back-end server. A back-end server need not contain a database that is subject to extraction processing.

On the other hand, each source Datareplicator can serve only one source HiRDB.

(2) Correspondence between source and target Datareplicators

To establish correspondence between source and target Datareplicators, you must design the items discussed below.

(a) Proportion

Source Datareplicator to target Datareplicators = 1 to m

Source Datareplicators to target Datareplicator = n to 1

Variable	UNIX	Windows
m	1 to $64^{\#}$	1 to $63^{\#}$
n	1 to 128	1 to 63

$\#$

If `sendmst` is specified in the `sendcontrol` operand, the value range is 1 to 4,096.

You can send update information from a single source Datareplicator to multiple target Datareplicators. If the HiRDB is a parallel server, you can send update information to multiple target Datareplicators per source Datareplicator at a server containing a back-end server.

On the other hand, a single target Datareplicator can receive update information from multiple source Datareplicators.

If you run multiple target Datareplicators for one target HiRDB, you can receive update information from 128 or more (in Windows, 63 or more) source systems (data

linkage identifiers). However, the following problems might occur:

- Import processing might result in lock errors.
- The order in which the update information is imported might not be as expected.

(b) Target identifiers

The identifier used by the source Datareplicator to identify the destination of update information is called a target identifier. Specify all target identifiers in the `sendidxx` or `sendidxxxx` operand in the extraction system definition and specify a target identifier for transmission of specific update information in the transmission statement in the extraction definition.

You can identify the transmission target by creating a transmission environment definition for each target identifier and specifying the target identifier in the `sendhdsid` operand in the transmission environment definition. The source Datareplicator can start transmission processing or specify reduced-mode operation for each target identifier.

You must specify the `sendidxx` or `sendidxxxx` operands consecutively in ascending order. To add a `sendidxx` or `sendidxxxx` operand during operation, you must initialize the source and target environments. To provide for the possibility of a future increase in the number of destinations, specify absent numbers (by specifying `**` as the target identifier for each absent number) with the `sendidxx` or `sendidxxxx` operand. When you provide absent numbers, you can establish data linkage with a specified target simply by initializing the target (there is no need to initialize the entire environment at the source and target).

(c) Data linkage identifiers

The identifier used to establish correspondence between source and target systems for data linkage is called a data linkage identifier. Specify the same data linkage identifier in the source and target systems between which data is to be linked. A specified data linkage identifier must be unique in the target system. For a source Datareplicator, use the `dsid` operand in the extraction environment definition; for a target Datareplicator, use the `dsidxxx` operand in the import system definition.

If the source system is HiRDB, specify a data linkage identifier for each extraction unit. An extraction unit is a unit of update information extraction and transmission. When the source HiRDB is a single server, a single source Datareplicator achieves extraction and transmission; in this case, one source Datareplicator is equivalent to one extraction unit. On the other hand, when the source HiRDB is a parallel server, the source Datareplicator for each back-end server achieves extraction and transmission; in this case, the source Datareplicator for each back-end server is equivalent to one extraction unit.

You must specify the `dsidxxx` operands consecutively in ascending order. To add a `dsidxxx` operand during operation, you must initialize the source and target

environments. To provide for the possibility of a future increase in the number of destinations, specify absent numbers (by specifying `**` as the data linkage identifier for each absent number) with the `dsidxxx` operand. When you provide absent numbers, you can establish data linkage with a specified source simply by initializing the environment of the source to be added (there is no need to initialize the entire environment at the source and target).

(d) Source Datareplicator identifiers

The identifier used to identify a source Datareplicator is called a source Datareplicator identifier. Specify the source Datareplicator identifier in the `hdeid` operand in the extraction system definition.

(e) Target Datareplicator identifiers

The identifier used to identify a target Datareplicator is called a target Datareplicator identifier. Specify the target Datareplicator identifier in the `hdsid` operand in the import system definition.

At the source Datareplicator, use the `sendhdsid` operand in the transmission environment definition to specify the target Datareplicator identifier at the destination of update information.

(f) Service name

To transfer update information between source and target Datareplicators, you must add the name of the communications service in the following `services` file:

- UNIX Datareplicator: `/etc/services`
- Windows Datareplicator: `Windows-system-directory\drivers\etc\services`

Specify the name of the applicable communications service in the `hdeservices` operand in the transmission environment definition for the source Datareplicator.

(g) Host name

To send update information, you must register the host name in the following `hosts` file at the source system:

- UNIX Datareplicator: `/etc/hosts`
- Windows Datareplicator: `Windows-system-directory\drivers\etc\hosts`

Specify the host name in the `hdehost` operand in the transmission environment definition.

(3) Correspondence between target Datareplicator and target HiRDB

To establish correspondence between target Datareplicator and target HiRDB, you must design the items discussed below.

(a) Proportion

Target Datareplicators to target HiRDB = n to 1

There can be only one target HiRDB for each target Datareplicator. On the other hand, there can be multiple target Datareplicators per target HiRDB.

(b) HiRDB authorization identifier and password

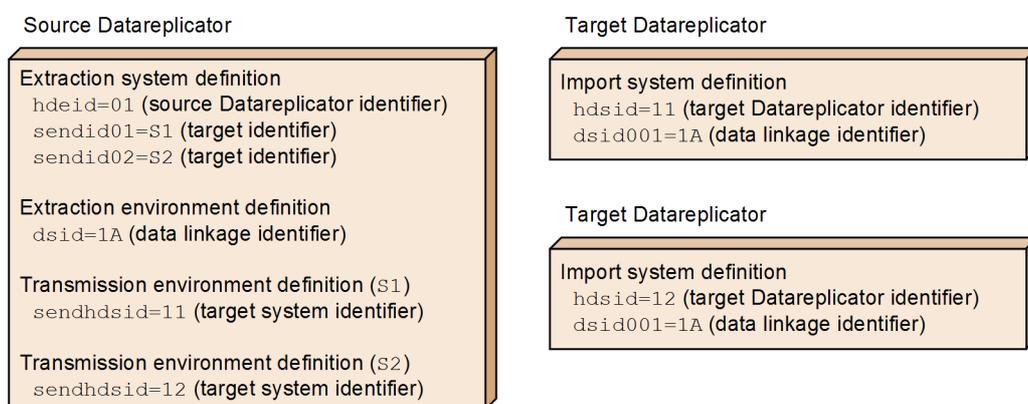
A target Datareplicator issues SQL statements and executes import processing in the form of a UAP that runs on the target HiRDB. Therefore, a HiRDB authorization identifier and password are required in order to establish connection with a target Datareplicator. To specify a HiRDB authorization identifier and password, use the `hirdbusr` operand in the import system definition. For details about the import system definition, see *5.8 Import system definition*.

You can also enter the password from the standard input when the target Datareplicator is started.

(4) Example definition of identifiers

The following figure shows an example definition of identifiers when data is linked from one source Datareplicator to two target Datareplicators.

Figure 4-17: Example definition of identifiers



Explanation:

- This example specifies the source Datareplicator identifier and the target Datareplicator identifier in the `hdeid` operand in the extraction system definition and in the `hdsid` operand in the import system definition, respectively.
- The example specifies the data linkage identifier in the `dsid` operand in the extraction environment definition and in the `hdsid` operand in the import system definition. Specify the same data linkage identifier for both source

and target systems that link data.

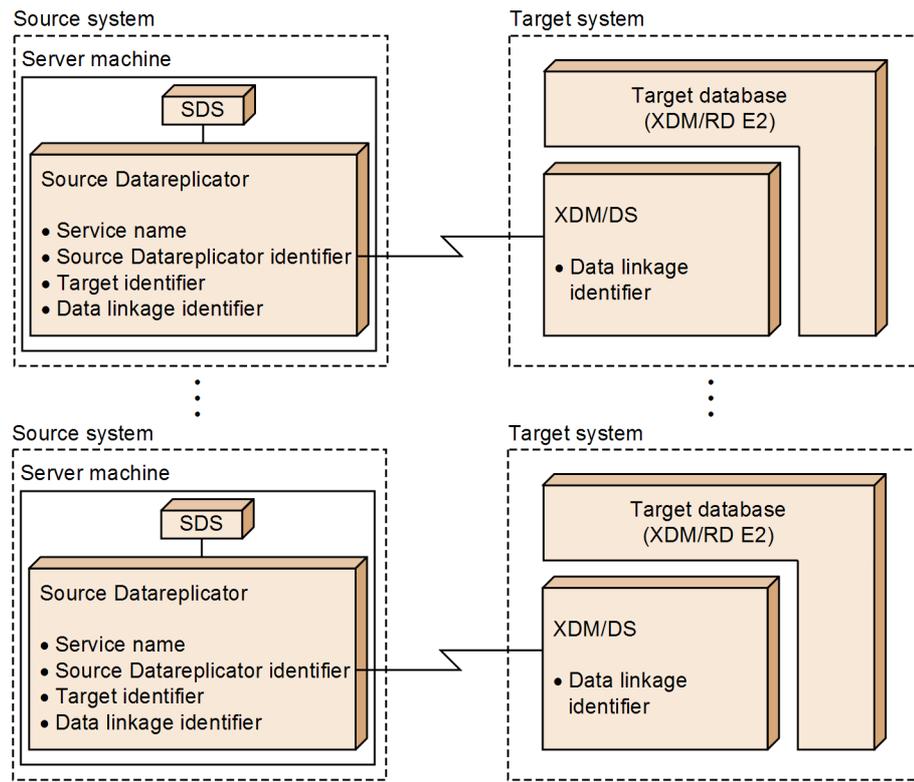
- The example specifies the target identifier in the `sendidxx` operand in the extraction system definition. Create a transmission environment definition for each target identifier, and then specify the target system identifier (target Datareplicator identifier) in the `sendhdsid` operand.

4.4.2 Designing data linkage from a HiRDB to a mainframe database

In designing data linkage from a HiRDB to a mainframe database (XDM/RD E2), the correspondence between the source and target systems depends on whether the source HiRDB is a single server or a parallel server.

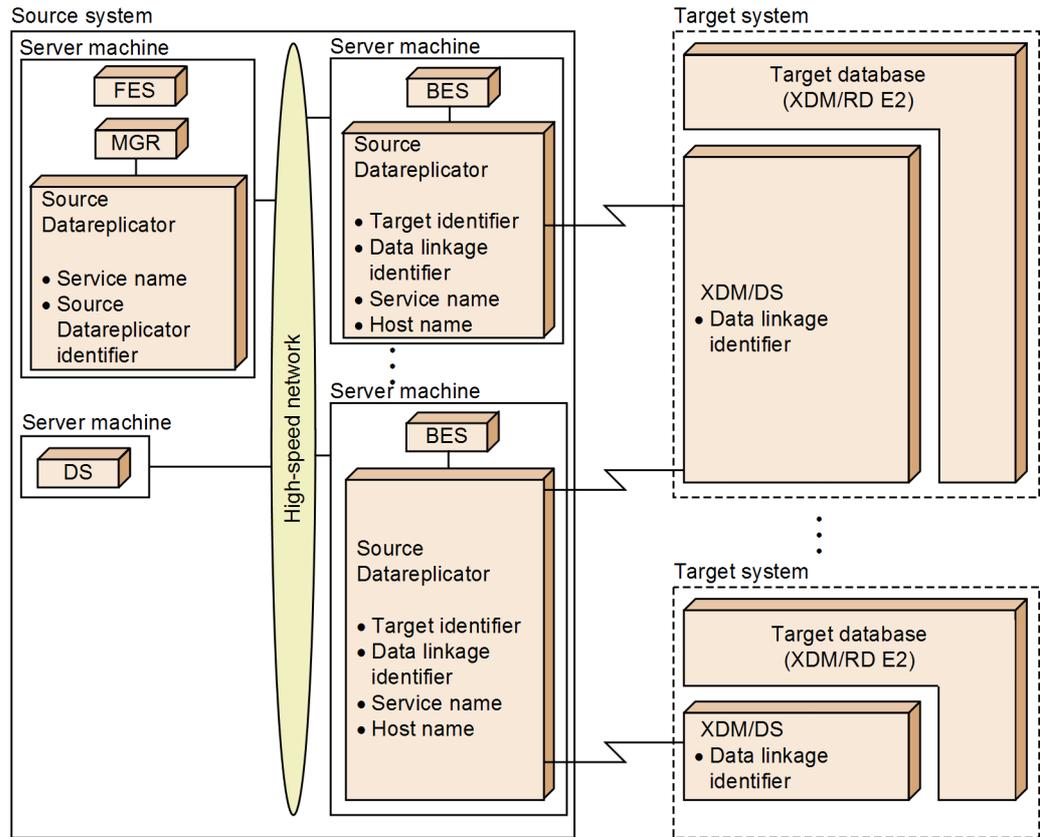
Figure 4-18 shows the correspondence when the source HiRDB is a single server, and Figure 4-19 shows the correspondence when the source HiRDB is a parallel server.

Figure 4-18: Correspondence between the source and target systems in designing data linkage from a HiRDB to a mainframe database: Source HiRDB is a single server



Legend:
 SDS: HiRDB single server

Figure 4-19: Correspondence between the source and target systems in designing data linkage from a HiRDB to a mainframe database: Source HiRDB is a parallel server



Legend:

- | | |
|------------------------------|-----------------------------|
| SDS : HiRDB single server | MGR : HiRDB system manager |
| FES : HiRDB front-end server | BES : HiRDB back-end server |
| DS : HiRDB dictionary server | |

(1) Correspondence between source HiRDB and source Datareplicator

For details about the correspondence between the source HiRDB and the source Datareplicator, see 4.4.1(1) *Correspondence between source HiRDB and source Datareplicator*.

(2) Correspondence between source Datareplicator and XDM/DS

You design the items discussed below to establish correspondence between the source Datareplicator and XDM/DS. This section explains the specifications for the target

Datareplicator; for details about the specifications of the XDM/DS items, see the manual *VOS3 XDM Data Linkage Facility XDM/DS Description and Definition*.

(a) Proportion

Source Datareplicator to XDM/DSs = 1 to m

Source Datareplicator to XDM/DS = 1 to 1

Variable	UNIX	Windows
m	1 to 64 [#]	1 to 63 [#]

#

If `sendmst` is specified in the `sendcontrol` operand, the value range is 1 to 4,096.

You can send update information from a single source Datareplicator to multiple target XDM/DSs. If the HiRDB is a parallel server, you can send update information to multiple target XDM/DSs per source Datareplicator at a server containing a back-end server.

On the other hand, one target XDM/DS can receive update information from only one source Datareplicator. If you run multiple XDM/DSs for one target XDM/DS, you can receive update information from multiple source systems (data linkage identifiers). However, the following problems might occur:

- Import processing might result in lock errors.
- The order in which the update information is imported might not be as expected.

(b) Target identifiers

For details about target identifiers, see *4.4.1(2)(b) Target identifiers*.

(c) Data linkage identifiers

For details about the data linkage identifiers, see *4.4.1(2)(c) Data linkage identifiers*.

(d) Source Datareplicator identifiers

The identifier used to identify a source Datareplicator is called a source Datareplicator identifier. Specify a source Datareplicator identifier in the `hdeid` operand in the extraction system definition.

(e) Target identifiers

For details about specifying target identifiers in XDM/DS, see the manual *VOS3 XDM Data Linkage Facility XDM/DS Description and Definition*.

(f) Service name

To transfer update information between source and target Datareplicators, you must

add the name of the communications service in the following `services` file:

- UNIX Datareplicator: `/etc/services`
- Windows Datareplicator: `Windows-system-directory\drivers\etc\services`

Specify the name of the applicable communications service in the `hdeservice` operand in the transmission environment definition for the source Datareplicator.

(g) Host name

To send update information, you must register the host name in the following `hosts` file at the source system:

- UNIX Datareplicator: `/etc/hosts`
- Windows Datareplicator: `Windows-system-directory\drivers\etc\hosts`

Specify the host name in the `hdehost` operand in the transmission environment definition.

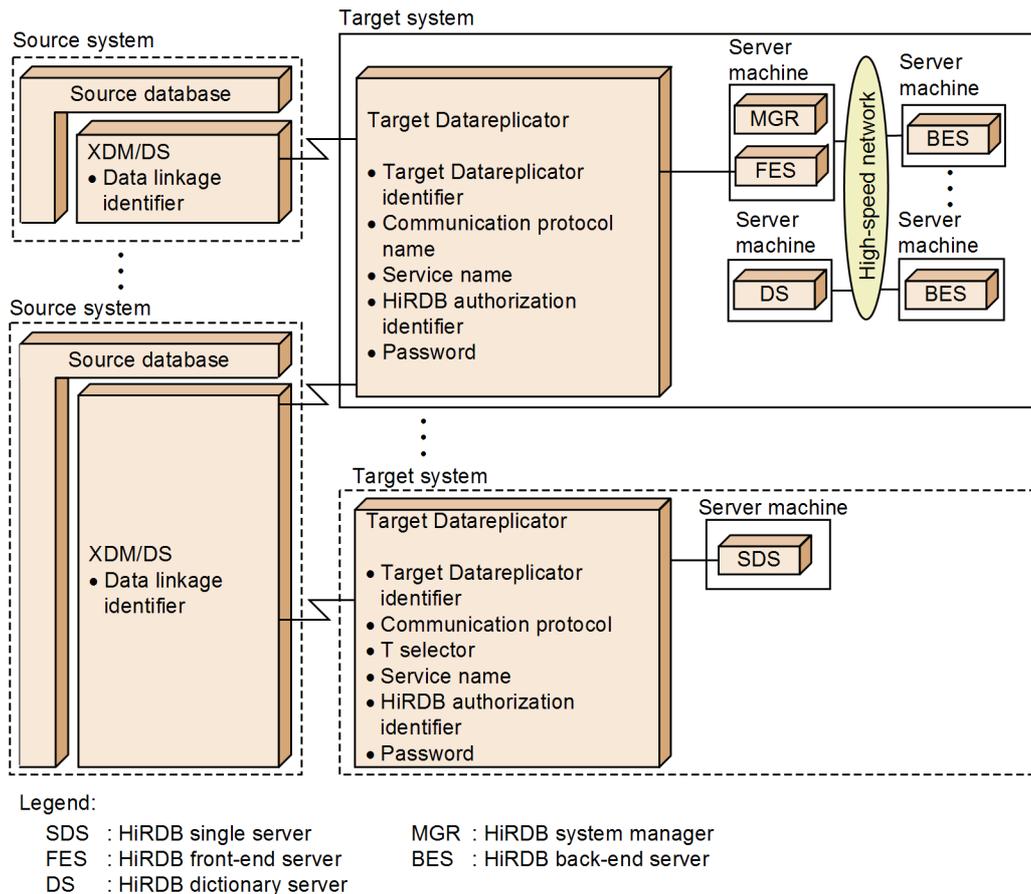
(3) Correspondence between XDM/DS and target database

For details about the correspondence between XDM/DS and a target database, see the manual *VOS3 XDM Data Linkage Facility XDM/DS Description and Definition*.

4.4.3 Designing data linkage from a mainframe database to a HiRDB

The following figure shows the correspondence between the source and target systems when data linkage is established from a mainframe database (XDM/SD E2, XDM/RD E2, ADM, PDMII E2, or TMS-4V/SP) to a HiRDB.

Figure 4-20: Correspondence between the source and target systems when data linkage is established from a mainframe database to a HiRDB



(1) Correspondence between source database and XDM/DS

For details about the correspondence between the source database and XDM/DS, see the *VOD3 XDM/DS* manual.

(2) Correspondence between XDM/DS and target Datareplicator

To establish correspondence between XDM/DS and the target Datareplicator, you must design the items discussed below. This section explains the specifications for the target Datareplicator; for details about the specification of each XDM/DS item, see the *VOS3 XDM/DS* manual.

(a) Proportion

XDM/DS to target Datarepliators = 1 to *m*

XDM/DSs to target Datareplicator = n to 1

Variable	UNIX	Windows
m	For details about the range of m , see the manual <i>VOS3 XDM Data Linkage Facility XDM/DS Description and Definition</i> .	
n	1 to 128	1 to 63

There can be multiple target Datareplicators per XDM/DS. This means that you can send update information from a single XDM/DS to multiple target Datareplicators.

On the other hand, one target Datareplicator can receive update information from multiple XDM/DSs.

If you run multiple target Datareplicators for one target HiRDB, you can receive update information from 128 or more (in Windows, 63 or more) source systems (data linkage identifiers). However, the following problems might occur:

- Import processing might result in lock errors.
- The order in which the update information is imported might not be as expected.

(b) Data linkage identifiers

For details about the data linkage identifiers, see *4.4.1(2)(c) Data linkage identifiers*.

(c) Target Datareplicator identifiers

The identifier used to identify a target Datareplicator is called a target Datareplicator identifier. Specify a target Datareplicator identifier in the `hdsid` operand in the import system definition.

At the source Datareplicator, use the `sendhdsid` operand in the transmission environment definition to specify the target Datareplicator identifier at the destination of the update information.

(d) Communications protocol

To transfer update information between XDM/DS and a target Datareplicator, you must specify the *communications protocol*. For the target Datareplicator, specify the `protocol1` or `protocol2` operand in the import system definition.

(e) T-selector

To transfer update information between XDM/DS and a target Datareplicator using the *OSI protocol (channel connection)*, you must specify a T-selector. For the target Datareplicator, use the `reflect_tselector` operand in the import system definition.

(f) Service name

To transfer update information between XDM/DS and a target Datareplicator using the

TCP/IP protocol, you must add the communication service name in the following services file:

- UNIX Datareplicator: `/etc/services`
- Windows Datareplicator: `Windows-system-directory\drivers\etc\services`

Specify the applicable communication service name in the target Datareplicator's `hdsservices` operand.

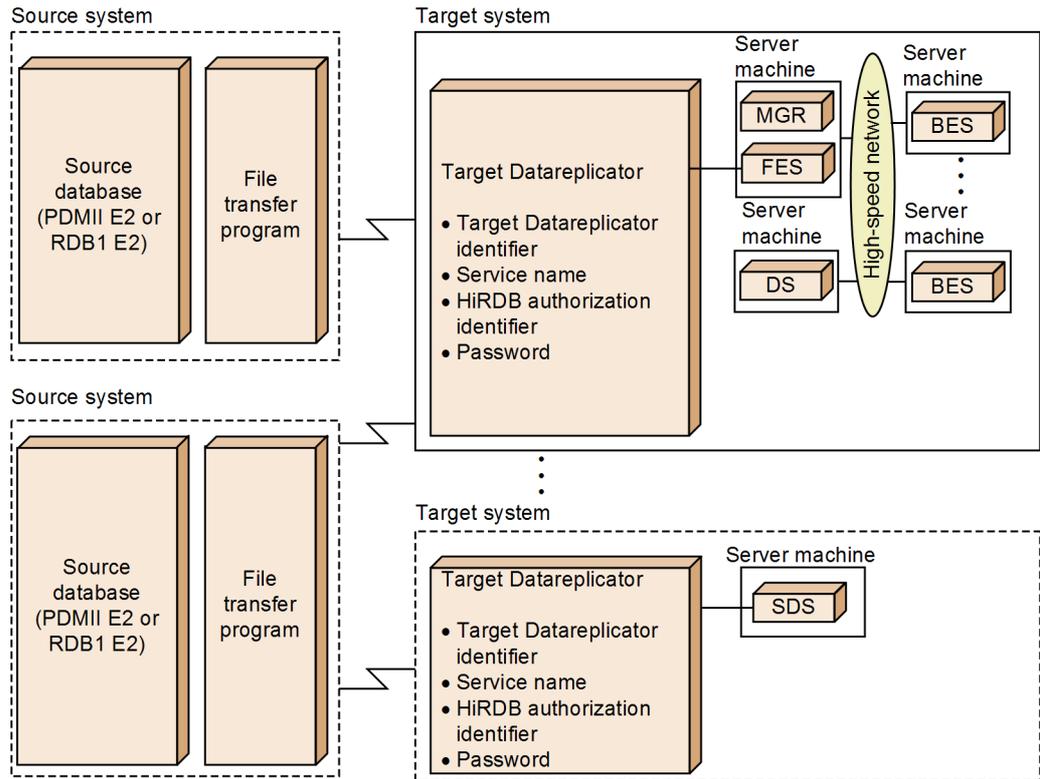
(3) Correspondence between target Datareplicator and target HiRDB

For details about the correspondence between the target Datareplicator and the target HiRDB, see *4.4.1(3) Correspondence between target Datareplicator and target HiRDB*.

4.4.4 Designing data linkage from a mainframe database to a HiRDB using SAM files

The figure below shows the correspondence between the source and target systems when data linkage is established from a mainframe database (PDMII E2 or RDB1 E2) to a HiRDB using a SAM file. No data linkage product is needed for the source database. However, a program is required in order to transfer the created SAM file to the target system.

Figure 4-21: Correspondence between the source and target systems when data linkage is established from a mainframe database (PDM2 E2 or RDB1 E2) to a HiRDB



Legend:

- | | |
|------------------------------|-----------------------------|
| SDS : HiRDB single server | MGR : HiRDB system manager |
| FES : HiRDB front-end server | BES : HiRDB back-end server |
| DS : HiRDB dictionary server | |

(1) Correspondence between mainframe database and target Datareplicator

To establish correspondence between a mainframe database that uses SAM files (PDM2 E2 or RDB1 R2) and a target Datareplicator, you must design the items discussed below. This section explains the specifications for the target Datareplicator.

(a) Proportion

There are no limitations on the ratio of mainframe databases to target Datareplicators. However, a single target Datareplicator can execute a maximum of 128 (UNIX Datareplicator) or 63 (Windows Datareplicator) update information input commands (hdssamqin) at one time.

If you run multiple target Datareplicators for one target HiRDB, you can receive update information from over 128 source systems (data linkage identifiers). However, when update information is imported from multiple target Datareplicators into a single target HiRDB (database), the following problems might occur:

- Import processing might result in lock errors.
- The order in which the update information is imported might not be as expected.

(b) Data linkage identifiers

For the target Datareplicator, specify the data linkage identifier in the `dsidxxx` operand in the import system definition. Specify this data linkage identifier in the `hdssamqin` command.

For details about the data linkage identifiers, see *4.4.1(2)(c) Data linkage identifiers*.

(c) Target Datareplicator identifiers

The identifier used to identify a target Datareplicator is called a target Datareplicator identifier. Specify a target Datareplicator identifier in the `hdsid` operand in the import system definition.

At the source Datareplicator, use the `sendhdsid` operand in the transmission environment definition to specify the target Datareplicator identifier at the destination of the update information.

(2) Correspondence between target Datareplicator and target HiRDB

For details about the correspondence between the target Datareplicator and the target HiRDB, see *4.4.1(3) Correspondence between target Datareplicator and target HiRDB*.

4.5 Designing the data linkage system mode

This section uses an example of data linkage from one HiRDB to another HiRDB to explain the design of the data linkage system mode.

4.5.1 Data linkage system modes

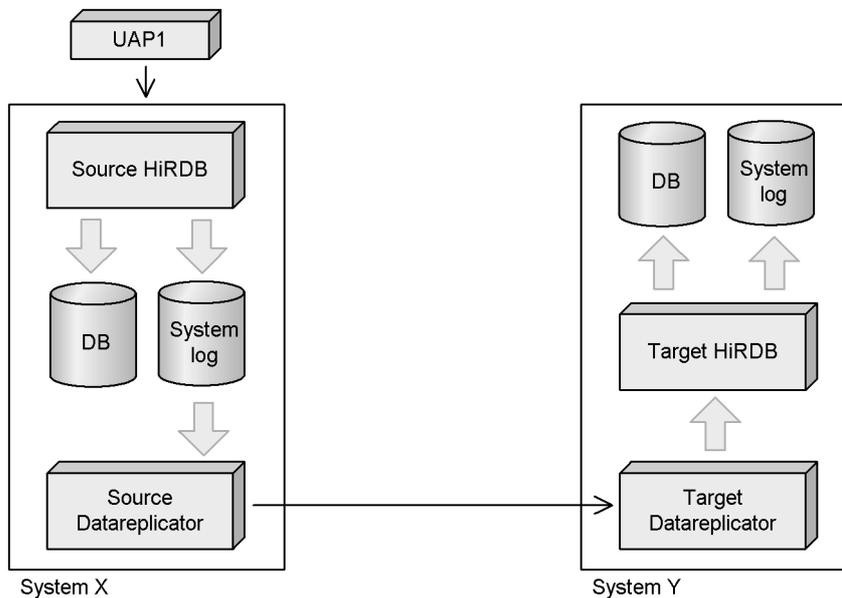
This section describes the types of data linkage systems you can design with Datareplicator.

(1) Unidirectional updating system

With a *unidirectional updating system*, one end is the source system and the other end is the target system. In such a system, data flows in one direction only.

The following figure shows an example of a unidirectional updating system.

Figure 4-22: Example of a unidirectional updating system

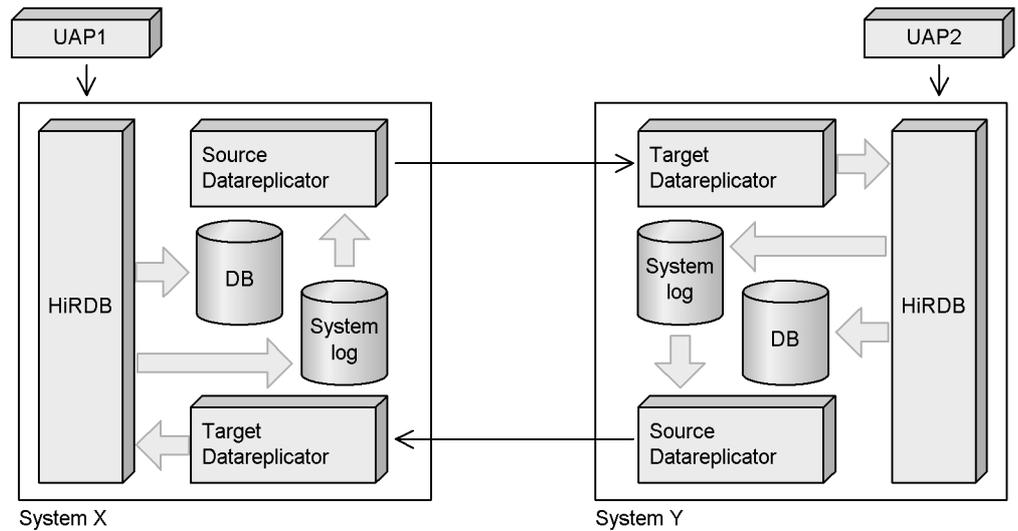


(2) Bidirectional updating system

With a *bidirectional updating system*, each end can be the source or target system. In such a system, data flows in both directions.

The following figure shows an example of a bidirectional updating system.

Figure 4-23: Example of a bidirectional updating system



4.5.2 Designing a data linkage system for application to a hierarchical system

This section explains the design of a data linkage system for application to a hierarchical system.

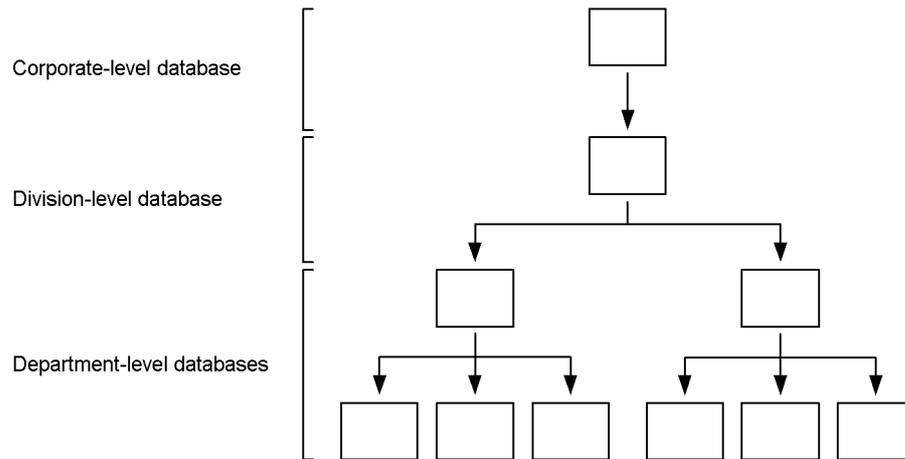
(1) Application of a unidirectional updating system to a hierarchical system

You can apply a unidirectional updating system to a hierarchical system with the following characteristic:

- Only information on updates to a higher-level database needs to be imported into lower-level databases. There is no need to import information on updates to a lower-level database into higher-level databases.

The following figure shows a hierarchical system that uses a unidirectional updating system.

Figure 4-24: Hierarchical system that uses a unidirectional updating system



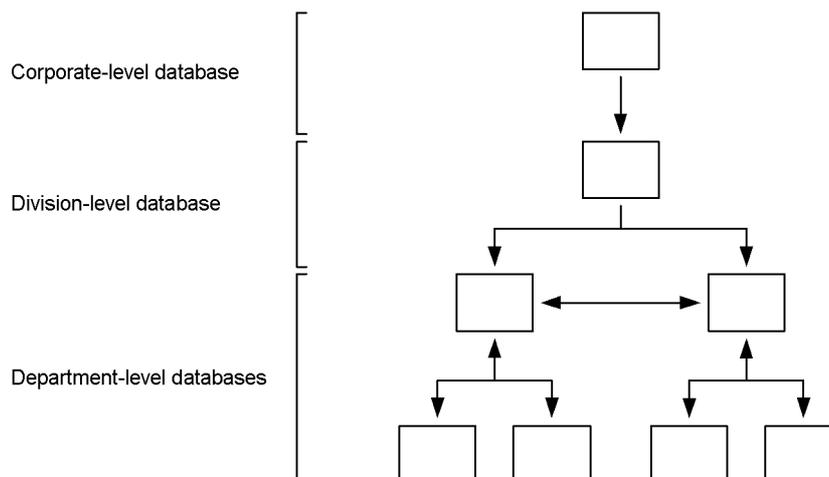
(2) Application of a bidirectional updating system to a hierarchical system

You can apply a bidirectional updating system to a hierarchical system with the following two characteristics:

- Only information that updates a large database, such as the corporate-level database or a division-level database, needs to be imported into lower-level databases; there is no need to import information on updates to a lower-level database into higher-level databases. In this case, the corporate-level and division-level databases use the unidirectional updating system.
- A database with limited information, such as a departmental-level database or a database at an even lower level, shares its information. In this case, the departmental-level and lower-level databases use the bidirectional updating system.

The following figure shows a hierarchical system that uses a bidirectional updating system.

Figure 4-25: Hierarchical system that uses a bidirectional updating system



4.5.3 Notes on a data linkage system that links multiple systems

This section discusses important topics concerning a data linkage system that links multiple systems.

(1) Updating the same target table from multiple source systems

If multiple source systems can update the same target table at the same time, it is impossible to identify the system that updated specific data. For example, if an all-entry UPDATE occurs on the same table at multiple source systems, the target system cannot identify the final system that updated specific data.

To avoid this, you can partition the target table by key ranges and limit the range of updating by each source system.

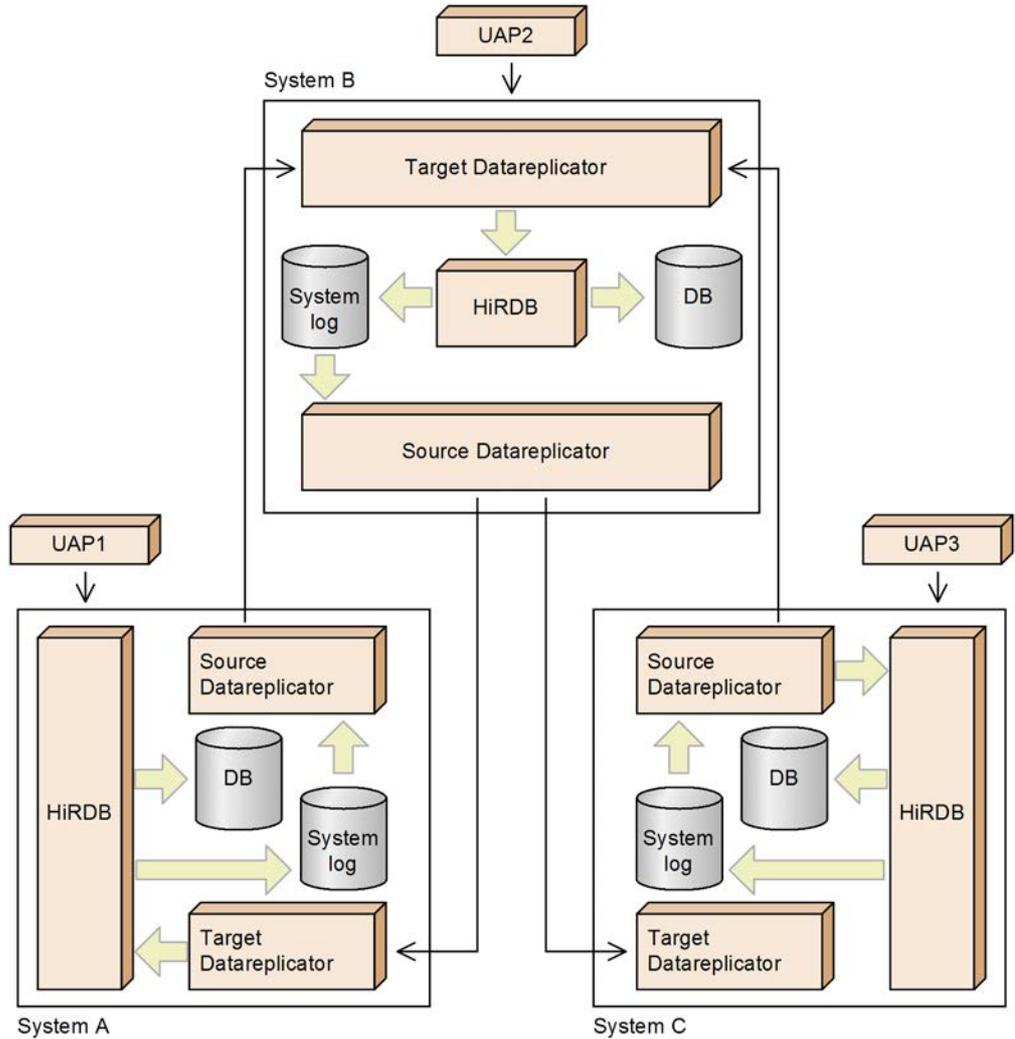
(2) Order of data linkage among multiple systems

If data linkage is established from one source system to multiple target systems, the order of import processing among the target systems is undefined because it depends on each target system's processing speed and workload (however, the order of transactions at the source system is guaranteed).

To define the order of data linkage among multiple systems, use a single line to connect all the systems. For example, if data is to be linked among three systems, A, B, and C, you can link the systems with a single line by first linking between systems A and B, and then linking between systems B and C.

The following figure shows an example system in which multiple systems are connected by a single line.

Figure 4-26: Example system in which multiple systems are connected by a single line



(3) Suppression of loopback

A bidirectional updating system transfers update information among multiple systems. After some update information has been imported, it might be extracted again and sent back to the original sender, resulting in an endless repetition of extractions and imports of the same update information among multiple systems. This is called *loopback*.

To avoid loopback, specify the identifier of the update information's original receiver, transmission to which is to be suppressed (transmission-suppressed original receiver

identifier) for each target identifier; you make this specification in the `nsndidxxx` operand in the transmission environment definition (where `xxx` is an integer in the range 001 to 256). For the transmission-suppressed original receiver identifier, specify the identifier of the source Datareplicator in the data linkage system that is to be subject to suppression. For example, in data linkage system A (`hdeid=01`, `hdsid=02`), to suppress update information from being sent from data linkage system B (`hdeid=11`, `hdsid=12`), specify `nsndid001=11` in the transmission environment definition for data linkage system A.

When you specify transmission suppression, you cannot use a UAP name that begins with `hdssqle` (otherwise, while transmission suppression is in effect, transmission of update information by a UAP whose name begins with `hdssqle` might be suppressed).

The following figure shows an example of a bidirectional updating system with loopback suppression.

Datareplicator processing

Flow of UAP1's update information:

1. At system X, UAP1's update information is stored in the system log file.
2. At system X, the Datareplicator uses the extraction facility to extract UAP1's update information and sends it to system Y (suppression is specified at system X for system Y's `nsndid001=02`, but UAP1's update information is not suppressed at this stage because it has no original receiver).
3. At system Y, the Datareplicator uses the import facility to import the update information that was sent in 2 above and stores it in the system log file.
4. At system Y, the Datareplicator uses the extraction facility to extract the update information in 3 above and sends it to system Z (suppression is specified at system Y for system X's `nsndid001=01` and system Z's `nsndid001=03`; in this case, only transmission to system X is suppressed, because the original receiver of UAP1's update information is system X at this stage).
5. At system Z, the Datareplicator uses the import facility to import the update information that was sent in 4 above and stores it in the system log file.
6. At system Z, the Datareplicator uses the extraction facility to extract the update information in 5 above but does not send it (suppression is specified at system Z for system Y's `nsndid001=02`; in this case, only transmission to system Y is suppressed, because the original receiver of UAP1's update information is system Y at this stage).

Flow of UAP2's update information:

1. At system Y, UAP2's update information is stored in the system log file.
2. At system Y, the Datareplicator uses the extraction facility to extract UAP2's update information and sends it to systems X and Z (suppression is specified at system Y for system X's `nsndid001=01` and system Z's `nsndid001=03`, but UAP2's update information is not suppressed at this stage because it has no original receiver).
3. At systems X and Z, the Datareplicators use the import facility to import the update information sent in 2 above and store it in the system log files.
4. At systems X and Z, the Datareplicators use the extraction facility to extract the update information in 3 above, but do not send it (suppression is specified at systems X and Z for system Y's `nsndid001=02`; in this case, only transmission to system Y is suppressed, because the original receiver of UAP2's update information is system Y at this stage).

Flow of UAP3's update information:

1. At system Z, UAP3's update information is stored in the system log file

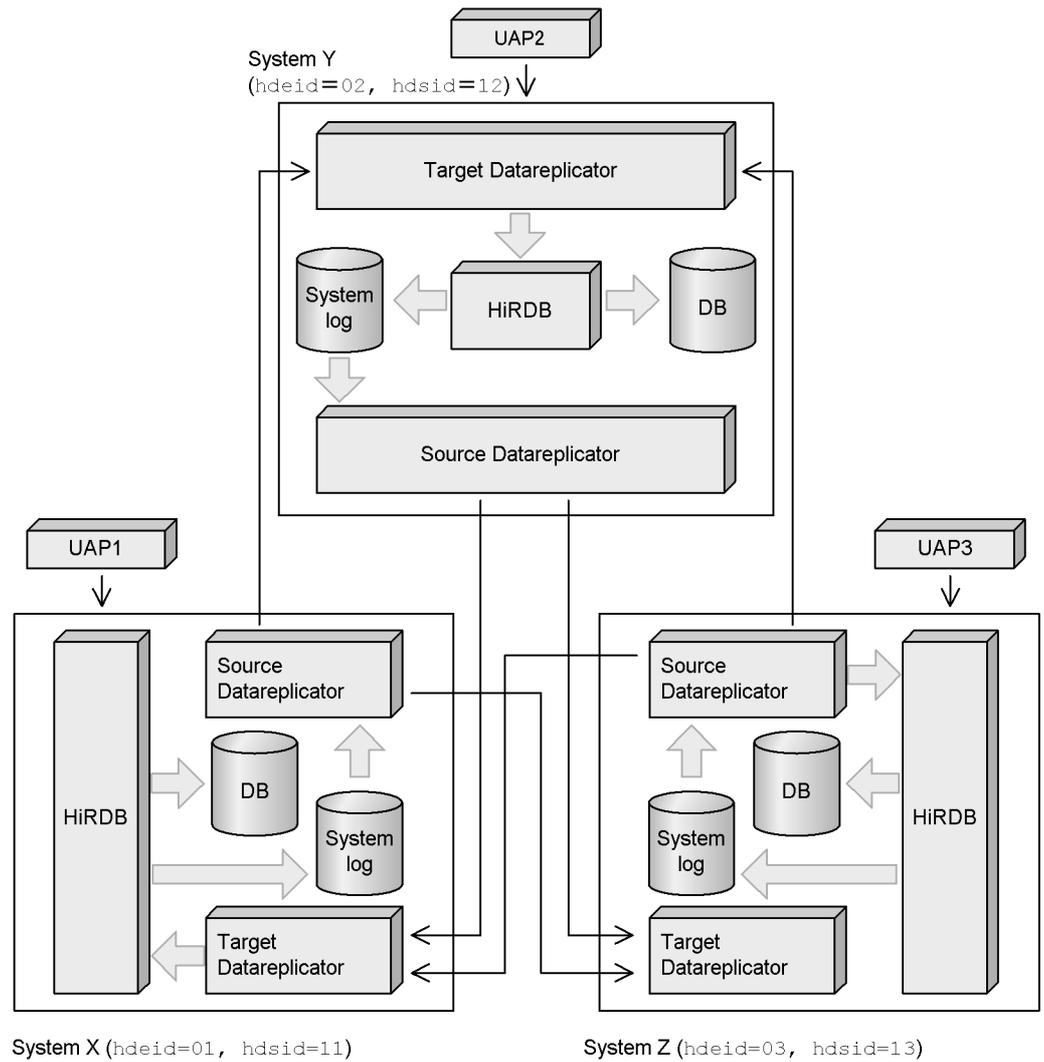
4. System Design

2. At system Z, the Datareplicator uses the extraction facility to extract UAP3's update information and sends it to system Y (suppression is specified at system Z for system Y's `nsndid001=02`, but UAP3's update information is not suppressed at this stage because it has no original receiver).
3. At system Y, the Datareplicator uses the import facility to import the update information that was sent in 2 above and stores it in the system log file.
4. At system Y, the Datareplicator uses the extraction facility to extract the update information in 3 above and sends it to system Z (suppression is specified at system Y for system X's `nsndid001=01` and system Z's `nsndid001=03`; in this case, only transmission to system Z is suppressed, because the original receiver of UAP3's update information is system Z at this stage).
5. At system X, the Datareplicator uses the import facility to import the update information that was sent in 4 above and stores it in the system log file.
6. At system X, the Datareplicator uses the extraction facility to extract the update information in 5 above, but does not send it (suppression is specified at system X for system Y's `nsndid001=02`; in this case, only transmission to system Y is suppressed, because the original receiver of UAP3's update information is system Y at this stage).

(4) Data linkage system conducting bidirectional updating among more than two systems

The following figure shows an example of bidirectional updating among more than two systems.

Figure 4-28: Example of bidirectional updating among more than two systems



Specifications:

System X: sendhdsid=12, nsendid001=02, nsendid002=03
 sendhdsid=13, nsendid001=02, nsendid002=03

System Y: sendhdsid=11, nsendid001=01, nsendid002=03
 sendhdsid=13, nsendid001=01, nsendid002=03

System Z: sendhdsid=11, nsendid001=01, nsendid002=02
 sendhdsid=12, nsendid001=01, nsendid002=02

The following describes the Datareplicator processing in Figure 4-28.

Datareplicator processing

Flow of UAP1's update information:

1. At system X, UAP1's update information is stored in the system log file.
2. At system X, the Datareplicator uses the extraction facility to extract UAP1's update information and sends it to systems Y and Z (suppression is specified at system X, but UAP1's update information is not suppressed at this stage because it has no original receiver).
3. At systems Y and Z, the Datareplicators use the import facility to import the update information sent in 2 above and store it in the system log files.
4. At system Y, the Datareplicator does not send the update information in 3 above because suppression is specified for system X's and Z's `nsndid001=01` when system X is the extraction source.
5. Similarly at system Z, the Datareplicator does not send the update information in 3.

Loopback control also takes effect in the same manner for UAP2 and UAP3.

(5) Loopback suppression when a data linkage facility from XDM/DS is used

The following table shows the handling of update information sent from XDM/DS:

XDM/DS version	Handling of update information sent from XDM/DS	
	Loopback control used	Importing with a UOC routine
07-00 or earlier	Specify 00 in the <code>nsndidxxx</code> operand in the transmission environment definition. Otherwise, the update information will be sent to another system.	00 is specified for the interface block's (UINTERFACE_BLK) source Datareplicator identifier.
07-01 or later	For the <code>nsndidxxx</code> operand in the transmission environment definition, specify the character codes (EBCDIK) specified in the XDM/DS identifier clause of the XDM/DS startup definition. Otherwise, the update information will be sent to another system.	The character codes (EBCDIK) specified in the XDM/DS identifier clause of the XDM/DS startup definition are specified as the interface block's (UINTERFACE_BLK) source Datareplicator identifier.

4.6 Designing a source Datareplicator

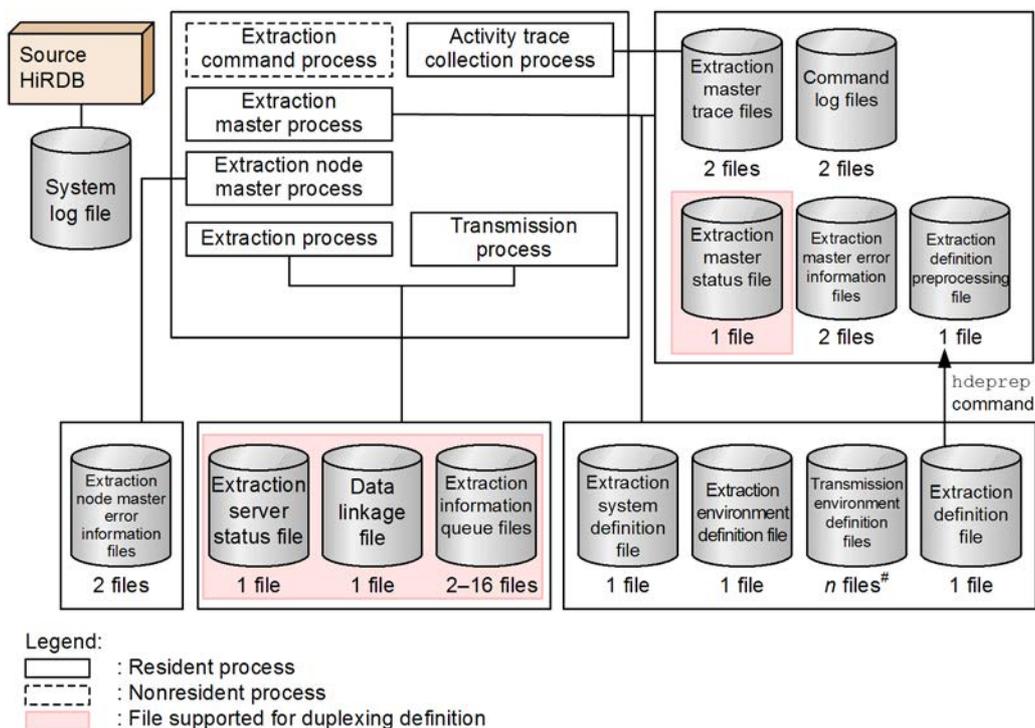
This section explains the system design for a source Datareplicator.

4.6.1 Source Datareplicator's file organization

Figures 4-29 and 4-30 show the source Datareplicator's file organizations.

For details about the source Datareplicator's directory structure in UNIX, see 2.3.2 *Directory structure of a source Datareplicator*; for details about the source Datareplicator's directory structure in Windows, see 2.7.2 *Directory structure of a source Datareplicator*.

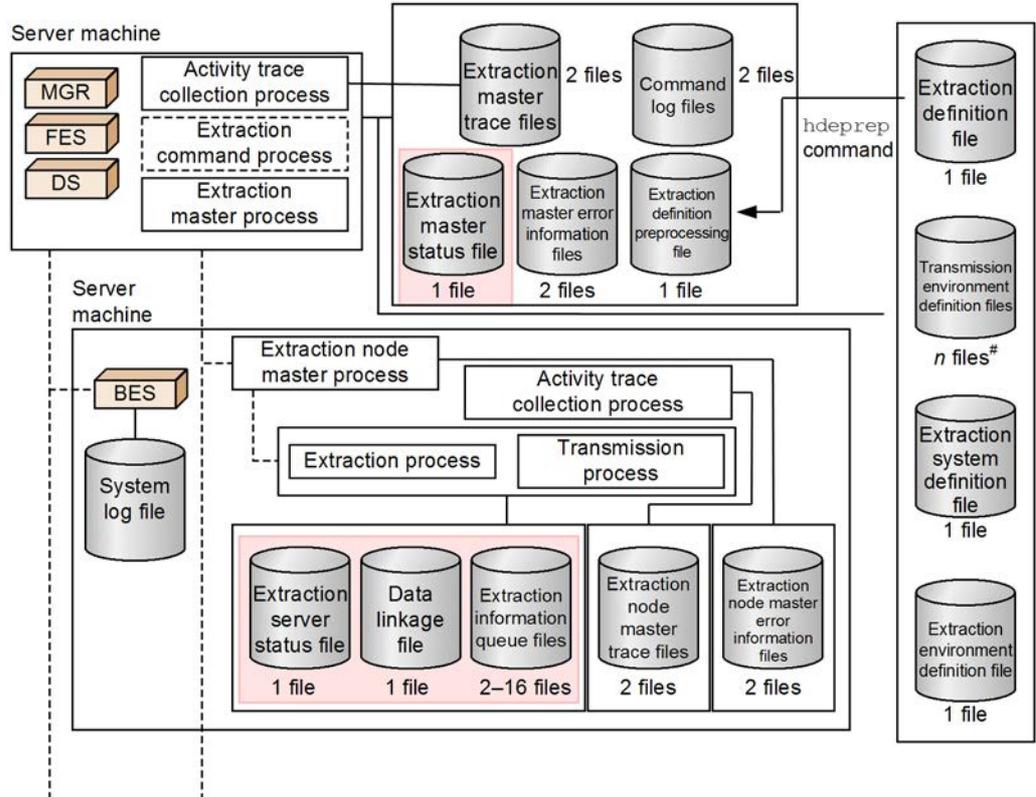
Figure 4-29: Source Datareplicator's file organization: Source HiRDB is a single server

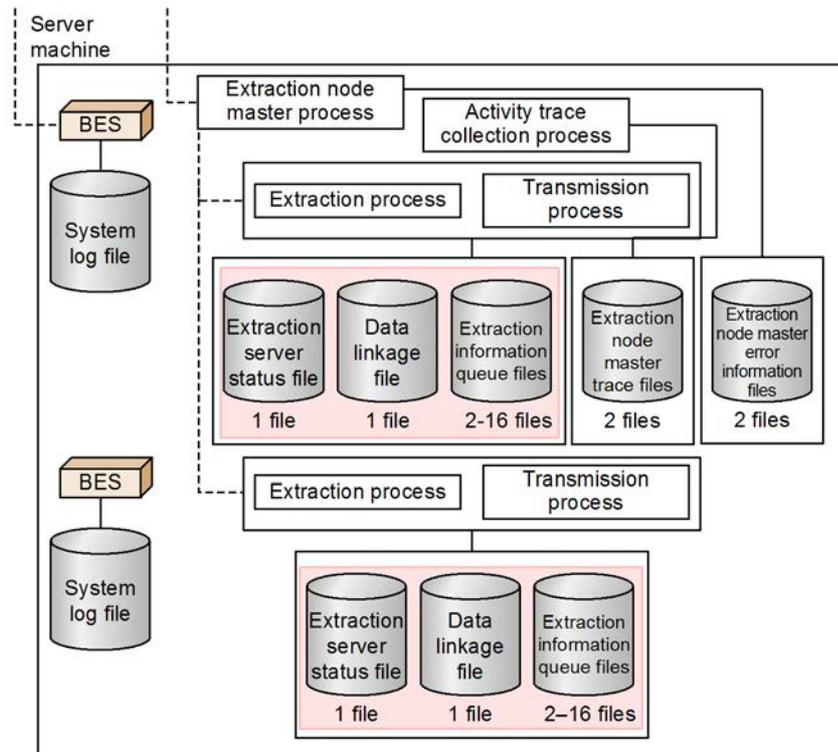


#

n indicates the number of target systems at the destination.

Figure 4-30: Source Datareplicator's file organization: Source HiRDB is a parallel server





Legend:

- MGR : HiRDB system manager
- DS : HiRDB dictionary server
- [] : Resident process
- [] : File supported for duplexing definition
- FES : HiRDB front-end server
- BES : HiRDB back-end server
- [] : Nonresident process
- [] : Control

#

n indicates the number of target systems at the destination.

4.6.2 Preparation of the files used with the source Datareplicator

This section explains the preparation of the following files, which are needed to use the source Datareplicator:

Files the user must create	Files created by the source Datareplicator during initial startup
<ul style="list-style-type: none"> • Extraction system definition file • Extraction environment definition file • Transmission environment definition files • Extraction definition file • Duplexing definition file 	<ul style="list-style-type: none"> • Extraction definition preprocessing file • Extraction information queue files • Extraction master status file • Extraction server status file • Extraction master error information files • Extraction node master error information files • Extraction master trace file • Extraction node master trace files • Data linkage file • Command log files • Duplexing control file

For details about the files, see *3.2.2 Files and processes used during extraction processing*.

For details about the handling of the files, see *6.4.2 Handling of the files used with the source Datareplicator*.

The following table provides information about creating the files used with the source Datareplicator.

Table 4-32: Creating the files used with the source Datareplicator

Filename		File type ^{#7}		Number of files	Required/ optional
		R ^{#8}	C ^{#9}		
Definition files	Extraction system definition file ^{#1}	Y	N	1 per source system (1 per MGR if the source HiRDB is a parallel server ^{#10})	Required
	Extraction environment definition file ^{#1}	Y	N	1 per source system (1 per MGR if the source HiRDB is a parallel server ^{#11})	Required
	Transmission environment definition file ^{#1}	Y	N	1 per target identifier (1 per MGR for each target identifier if the source HiRDB is a parallel server ^{#11})	Required
	Extraction definition file ^{#2}	Y	N	1 per source system (1 per MGR if the source HiRDB is a parallel server)	Required
	Duplexing definition file ^{#1}	Y	N	1 per source system (1 per MGR if the source HiRDB is a parallel server)	Optional
Extraction definition preprocessing file ^{#3}		Y	Y	1 per source system (1 per MGR if the source HiRDB is a parallel server)	Required
Extraction information queue files ^{#4, #12}		Y	Y	2 to 16 per source system (2 to 16 per BES if the source HiRDB is a parallel server)	Required
Status files ^{#4, #12}	Extraction master status file	Y	Y	1 per source system (1 per MGR if the source HiRDB is a parallel server)	Required
	Extraction server status file	Y	Y	1 per source system (1 per BES if the source HiRDB is a parallel server)	Required

4. System Design

Filename		File type ^{#7}		Number of files	Required/ optional
		R ^{#8}	C ^{#9}		
Error information files ^{#5, #6}	Extraction master error information files	Y	N	2 per source system (2 per MGR if the source HiRDB is a parallel server)	Required
	Extraction node master error information files	Y	N	2 per source system (2 per server machine containing a BES if the source HiRDB is a parallel server)	Required
Activity trace files	Extraction master trace files	Y	N	2 per source system	Optional
	Extraction node master trace files	Y	N	2 per source system (2 per BES if the source HiRDB is a parallel server)	Optional
Data linkage file ^{#4, #12}		Y	Y	1 per source system (1 per BES if the source HiRDB is a parallel server)	Required
Command log files ^{#6}		Y	N	2 per source system	Optional
Duplexing control file ^{#5}		Y	N	1 per source system	Optional

Legend:

MGR: System manager

BES: Back-end server (includes back-end servers that do not contain any databases subject to extraction processing)

R: UNIX regular file or Windows file

C: UNIX character special file

Y: Can be created.

N: Cannot be created.

#1

Use an OS editor to create this file before starting the source Datareplicator.

#2

Use an OS editor to create this file before executing the `hdeprep` command.

#3

To create this file, execute the `hdeprep` command after you have created the extraction definition file but before you start the source Datareplicator. The extraction definition preprocessing file is created automatically when the `hdeprep` command is executed. If the file is a character special file for UNIX, create a symbolic link to the character special file before you execute the `hdeprep` command.

#4

Before you start the source Datareplicator, execute the `hdestart -i` command to initialize the source Datareplicator. These files are created automatically when the `hdestart -i` command is executed. If the files are character special files for UNIX, create a symbolic link to the character special files before you start the source Datareplicator.

#5

These files are created automatically when the source Datareplicator is initialized.

#6

These files are created automatically when the source Datareplicator is started.

#7

Use the same file type for all the following files:

- Extraction information queue files
- Extraction server status file
- Data linkage file

#8

If the user creates the file (definition file), grant the `read` privilege to source Datareplicator users. Grant the `write` privilege as appropriate. For a file that is not created by the user (definition file), do not change the privilege.

#9

An OS command is used to create character special files. If you use a character special file, set the following privileges:

Data linkage file:

Grant the `read` and `write` privileges to source Datareplicator users and groups.

File other than data linkage file:

Grant the `read` and `write` privileges to source Datareplicator users.

If you use the system switchover facility, create the files in the character special file format.

When you configure the environment, we recommend that you execute the `hdestart -i` command with `init` specified to check file capacity so that a shortage of file capacity will not occur while the source Datareplicator is running.

#10

If the source HiRDB is a parallel server and different settings are used in HiRDB's environment variables (such as `PDDIR`, `PDCONFPATH`, and `SHLIB_PATH`) for each server machine, you can also create operands for the environment variables separately for each server machine.

#11

If the source HiRDB is a parallel server, you can also create this file for each BES.

#12

In UNIX, if the file type is OS regular files, data might not be output in the event of a system failure. Because the source Datareplicator uses the extraction information queue file, extraction status file, and data linkage file during error recovery, a failure cannot be recovered if no data has been output to these files. We recommend that you use character special files that have high reliability for the extraction information queue file, extraction status file, and data linkage file.

The following table describes the settings for the files used at the source Datareplicator.

Table 4-33: File settings used at the source Datareplicator

File type	File name	Setting
Extraction system definition file	<code>\$HDEPATH/hdeenv</code>	<ul style="list-style-type: none"> For the information to be defined, see 5.2 <i>Extraction system definition</i>.
Extraction environment definition file	<i>any-directory/any-name</i>	<ul style="list-style-type: none"> For the information to be defined, see 5.3 <i>Extraction environment definition</i>. Specify <i>any-name</i> in the <code>extdef</code> operand in the extraction system definition.
Transmission environment definition file	<i>any-directory/any-name</i>	<ul style="list-style-type: none"> For the information to be defined, see 5.4 <i>Transmission environment definition</i>. Specify <i>any-name</i> in the <code>senddef01</code> through <code>senddef64</code> operands in the extraction system definition.

File type	File name	Setting
Extraction definition file	<i>any-directory/ any-name</i>	<ul style="list-style-type: none"> For the information to be defined, see 5.5 <i>Extraction definition</i>. Specify <i>any-name</i> when you execute the <code>hdeprep</code> command and create the extraction definition preprocessing file.
Duplexing definition file	<i>any-directory/ any-name</i>	<ul style="list-style-type: none"> For the information to be defined, see 5.7 <i>Duplexing definition (source)</i>. Specify <i>any-name</i> in the <code>file_dupenv</code> operand in the extraction system definition file.
Extraction definition preprocessing file	<code>\$HDEPATH/ hde_prpfile</code>	--
Extraction information queue file #	<i>any-directory/ any-name_server-name</i>	<ul style="list-style-type: none"> Specify <i>any-name</i> in the <code>qufile001</code> through <code>qufile016</code> operands in the extraction environment definition. The specified name appended with <code>_server-name</code> becomes the file name. Make sure that the file name is unique in the source system. If the source HiRDB is a parallel server, make sure that the file name is unique on each back-end server. Specify the file size in the <code>queuesize</code> operand in the extraction environment definition.
Extraction master status file #	<code>\$HDEPATH/ mststatus</code>	--
Extraction server status file #	<code>\$HDEPATH/ sts_server-name</code>	--
Extraction master error information file	<code>\$HDEPATH/ msterrfile1 \$HDEPATH/ msterrfile2</code>	<ul style="list-style-type: none"> Specify the file size in the <code>errfilesz</code> operand in the extraction system definition.
Extraction node master error information file	<code>\$HDEPATH/ errfile1 \$HDEPATH/ errfile2</code>	<ul style="list-style-type: none"> If <code>true</code> is specified in the <code>errfile_unique</code> operand in the extraction system definition, <code>errfile1_host-name</code> and <code>errfile2_host-name</code> become the file names. Specify the file size in the <code>errfilesz</code> operand in the extraction system definition.

File type	File name	Setting
Extraction master trace file	\$HDEPATH/ msttrc.trc1 \$HDEPATH/ msttrc.trc2	<ul style="list-style-type: none"> Specify the file size in the <code>int_trc_trcfilesz</code> operand in the extraction system definition. The file can be edited and referenced by using the <code>hdstrcredit</code> command.
Extraction node master trace file	\$HDEPATH/ exttrc.trc1 \$HDEPATH/ exttrc.trc2	<ul style="list-style-type: none"> Specify the file size in the <code>int_trc_trcfilesz</code> operand in the extraction system definition. If <code>true</code> is specified in the <code>errfile_unique</code> operand in the extraction system definition, <code>exttrc.trc1_host-name</code> and <code>exttrc.trc2_host-name</code> become the file names. The file can be edited and referenced by using the <code>hdstrcredit</code> command.
Data linkage file [#]	\$HDEPATH/ hde_server-name	--
Command log file	<i>any-directory/ any-name_server-na me</i>	--
Duplexing control file	\$HDEPATH/ hde_fileenv.prp	--

Legend:

--: Not applicable

#

The file can be duplexed. However, if a file system area is used, the file cannot be duplexed.

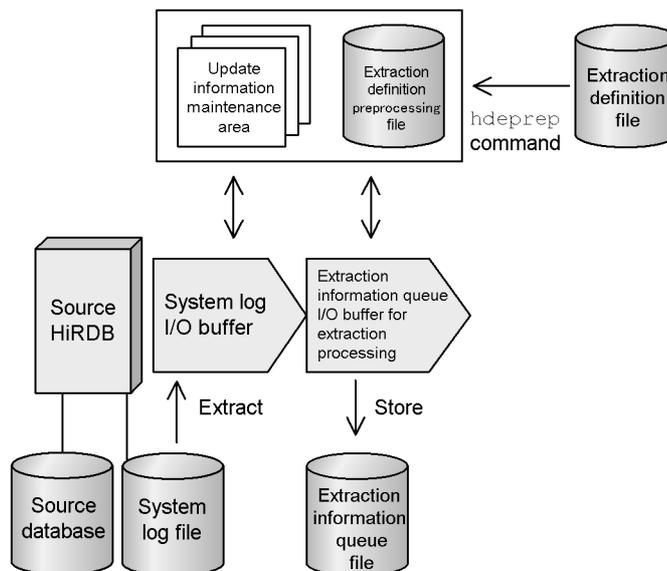
4.6.3 Designing the extraction procedure

This section explains the design of the extraction procedure.

(1) Designing the buffers to be used for extraction

The source Datareplicator uses a *system log I/O buffer* to read system log information from the system log file. The source Datareplicator takes from the system log information it has just read only the update information for the table subject to extraction processing, as specified in the extraction definition, and stores this information in an extraction information queue file. This is the use the source Datareplicator makes of the system log I/O buffer. The following figure provides an overview of the extraction procedure.

Figure 4-31: Overview of the extraction procedure



(a) System log I/O buffer

When the source Datareplicator starts, it allocates a *system log I/O buffer* in local memory. This section discusses how you estimate and specify the size of the system log I/O buffer.

Estimating the size of the system log I/O buffer

- The system log I/O buffer is used to extract system log information from the system log file. As the size of the system log I/O buffer increases, the number of read/write operations decreases, because more update information can be extracted from the system log file at one time. However, if the source Datareplicator terminates abnormally, restart processing requires more time because more update information must be extracted again during the restart.
- The size of the system log I/O buffer must be at least the value specified in `pd_log_max_data_size` for the source HiRDB.
- If the source HiRDB is a parallel server, you must estimate a system log I/O buffer size for each back-end server.

Specifying the size of the system log I/O buffer

- You use the `logiosize` operand in the extraction environment definition to specify the size of the system log I/O buffer. Datareplicator extracts update information from the system log file based on this size.

(b) Extraction information queue I/O buffer for extraction

When the source Datareplicator starts, it allocates *extraction information queue I/O buffers* in local memory. This section discusses how you estimate and specify the size of the extraction information queue I/O buffer for extraction.

Estimating the size of the extraction information queue I/O buffer for extraction

- The source Datareplicator uses the extraction information queue I/O buffer for extraction to store extracted update information in an extraction information queue file. As the size of the extraction information queue I/O buffer for extraction increases, the number of read/write operations decreases, because more update information can be stored in the extraction information queue file at one time. However, if the source Datareplicator terminates abnormally, restart processing requires more time because more update information must be extracted again during the restart.
- The size of the extraction information queue I/O buffer for extraction must be at least 1 less than the value specified in the `queuesize` operand in the extraction environment definition.
- Avoid a specification that results in a remainder from the formula $(\text{queuesize operand value in the extraction environment definition} - 1) \div \text{specified value}$. If there is a remainder, the size of the space in the extraction information queue file equivalent to the remainder will not be used.
- If the source HiRDB is a parallel server, you must estimate an extraction information queue I/O buffer size for each back-end server.
- An extraction process uses one extraction information queue I/O buffer.

Specifying the size of the extraction information queue I/O buffer for extraction

- You use the `quiosize` operand in the extraction environment definition to specify the size of the extraction information queue I/O buffer for extraction. Datareplicator uses this size for allocating the extraction information queue I/O buffer for extraction and the extraction information queue I/O buffers for transmission. For details about the extraction information queue I/O buffers for transmission, see *4.6.4 Designing the transmission procedure*.

(2) Designing an extraction error monitoring interval

The source Datareplicator uses the extraction master process to monitor for errors in the extraction node master process, extraction process, and transmission process (transmission master process). Error detection is faster if you use a short error monitoring interval. However, if the error monitoring interval is too short, the workload required for communications with the server becomes high.

You use the `watchintvl` operand in the extraction system definition to specify the error monitoring interval.

(3) Designing the unit of extraction environment definition

If the source HiRDB is a parallel server, the source Datareplicator executes extraction processing for each back-end server. In such a case, the source Datareplicator's extraction environment definition can include the environment common to all back-end servers and the environment for a specific back-end server.

You define the common environment under `commondef` and a specific environment under `besdef (server-name)` in the extraction environment definition. If you specify the same operand in both `commondef` and `besdef (server-name)`, the specification in `besdef` is effective for the server specified in `besdef`.

4.6.4 Designing the transmission procedure

This section explains the procedure for the design of the transmission procedure.

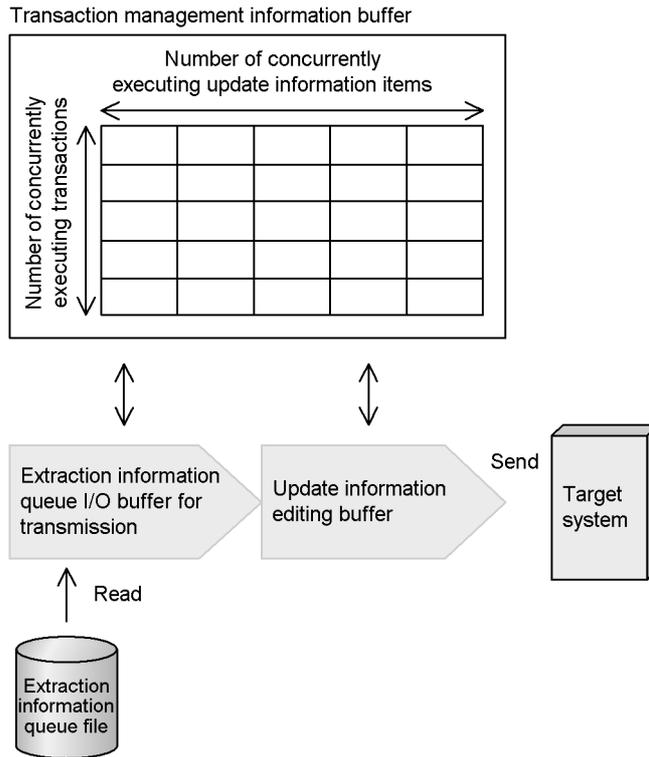
(1) Designing the buffers to be used for transmission

The source Datareplicator executes the following processing for each transmission target:

- The source Datareplicator uses extraction information queue I/O buffers for transmission to read update information that is subject to transmission from the extraction information queue file.
- Of all the update information read from the extraction information queue file, the source Datareplicator extracts and edits only the update information for completed transactions and sends this information to the target system. This is the use the source Datareplicator makes of the update information editing buffer.

Additionally, the source Datareplicator uses the *transaction management information buffer*, which contains transaction management information, to manage the transaction completion status. The following figure provides an overview of the transmission procedure.

Figure 4-32: Overview of the transmission procedure



(a) Transaction management information buffer

When the source Datareplicator starts, it allocates a *transaction management information buffer* in local memory. This section discusses how you estimate and specify the size of the transaction management information buffer.

Estimating the size of the transaction management information buffer

- The transaction management information buffer is used to store the transaction management information contained in the extraction information queue file. The source Datareplicator uses this transaction management information buffer for the purpose of editing and sending update information. If the size of the transaction management information becomes greater than the buffer size during data extraction or transmission, the source Datareplicator adds an area whose size is one-fifth of the initial value. Such buffer addition might increase the system workload and slow down the processing speed. As the buffer size increases, the probability of buffer addition decreases. However, if the buffer size is too large, a memory shortage might occur. Therefore, carefully estimate an appropriate size

for the transaction management information buffer so that neither area additions nor memory shortages occur.

- If the source HiRDB is a parallel server, you must estimate a transaction management information buffer size for each back-end server.

Specifying the size of the transaction management information buffer

- You use the `maxtran` and `maxtrandata` operands in the transmission environment definition to specify the size of the transaction management information buffer. The source Datareplicator multiplies the value of the `maxtran` operand, which is the maximum number of concurrently executable transactions, by the value of the `maxtrandata` operand, which is the maximum number of update information items in a transaction, and uses the resulting value for the initial size of the transaction management information buffer.

(b) Extraction information queue I/O buffers for transmission

When the source Datareplicator starts, it allocates *extraction information queue I/O buffers for transmission* in local memory. This section discusses how you estimate and specify the size and number of extraction information queue I/O buffers for transmission.

Estimating the size and number of extraction information queue I/O buffers for transmission

- The source Datareplicator uses extraction information queue I/O buffers for transmission to read update information stored in an extraction information queue file. It is necessary that the update information that is read be the same size as the stored update information that was extracted from the system log file. Therefore, an extraction information queue I/O buffer for transmission must be the same size as the extraction information queue I/O buffer for extraction. For details about the extraction information queue I/O buffer for extraction, see *4.6.3 Designing the extraction procedure*.
- You can specify the number of extraction information queue I/O buffers for transmission. If you have specified a transmission interval, the source Datareplicator reads update information at the specified transmission interval, and then sends it. If there are many buffers, the source Datareplicator can use the update information in the buffers, thereby reducing the number of read operations on the extraction information queue file.

Specifying the size of an extraction information queue I/O buffer for transmission

- The size of an extraction information queue I/O buffer for transmission is the same as the size of the extraction information queue I/O buffer for extraction. For details about the extraction information queue I/O buffer for extraction, see *4.6.3 Designing the extraction procedure*.

Specifying the number of extraction information queue I/O buffers for transmission

- You use the `readbufnum` operand in the transmission environment definition to specify the number of extraction information queue I/O buffers for transmission.

(c) Update information editing buffer

When the source Datareplicator starts, it allocates the *update information editing buffer* in local memory. This section discusses how you estimate and specify the size of the update information editing buffer.

Estimating the size of the update information editing buffer

- The source Datareplicator uses the update information editing buffer to edit update information. As the size of the update information editing buffer increases, more update information can be edited at one time, thereby reducing the number of read/write operations. However, if the update information editing buffer is too large, a memory shortage might occur.
- The update information editing buffer must be large enough to store at least one SQL update information item.
- If the source HiRDB is a parallel server, you must specify an update information editing buffer size for each back-end server.

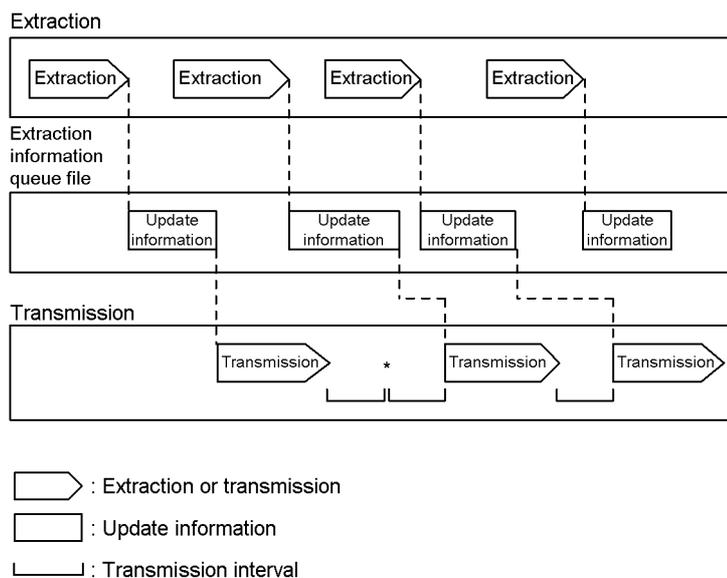
Specifying the size of the update information editing buffer

- You use the `editbufsize` operand in the transmission environment definition to specify the size of the update information editing buffer.

(2) Designing a transmission interval for transmission processing

The source Datareplicator sends the previous update information to the target in increments at a specified interval called the *transmission interval*. You use the `sendintvl` operand in the transmission environment definition to specify the transmission interval. The following figure provides an overview of the transmission interval for transmission processing.

Figure 4-33: Overview of transmission interval for transmission processing



* Update information is not sent during the first transmission interval, because extraction is not yet completed. The update information is sent during the second transmission interval, because extraction has been completed.

(3) Connection retries count

If connection establishment with the target system fails, the source Datareplicator can retry the connection establishment; this is called *connection retry*. You use the `retrynum` operand in the transmission environment definition to specify the maximum number of times the source Datareplicator is permitted to retry connection establishment. If no connection retries count is specified, the source Datareplicator waits for the next transmission interval and attempts to establish connection until connection establishment is successful.

(4) Designing reduced-mode transmission processing

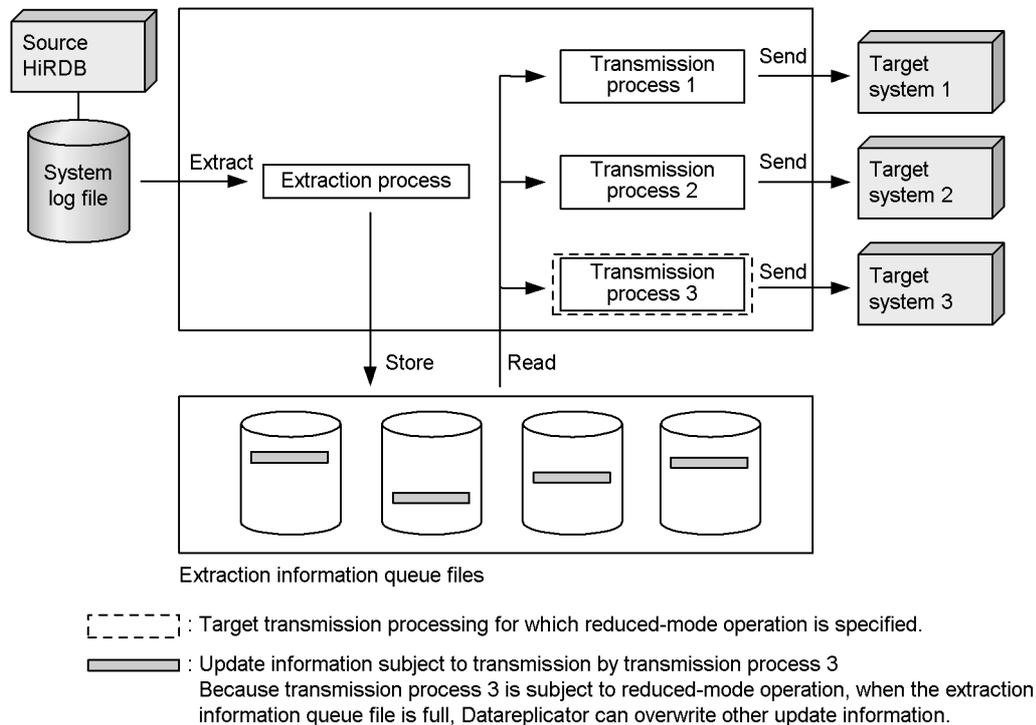
The source Datareplicator can send update information from one extraction environment to multiple destinations. However, if update information remains for some reason in the next extraction information queue file to be used, Datareplicator cannot swap extraction information queue files, making it impossible to extract the remaining update information from the system log file.

If the source Datareplicator cannot swap extraction information queue files, it cancels data linkage with the specified destination only, discards the corresponding update information from the next extraction information queue file to be used, and then swaps files. This is called *reduced-mode transmission processing*. You use the `overwrite`

operand in the transmission environment definition to specify use of reduced-mode transmission processing.

The following figure shows the concept of reduced-mode transmission processing.

Figure 4-34: Concept of reduced-mode transmission processing



(a) Operation when reduced-mode transmission processing is specified

If update information remains in the next extraction information queue file to be used while Datareplicator cannot swap extraction information queue files, and reduced-mode transmission processing is specified for that update information's destination, Datareplicator stops only the corresponding transmission process and discards that update information only from the next extraction information queue file to be used. As a result, Datareplicator can then swap extraction information queue files and resume extracting update information from the system log file.

When reduced-mode transmission processing occurs, Datareplicator does not guarantee conformity between the source database and the target database at the destination that is subject to reduced-mode transmission processing. Therefore, you might have to re-create the target database at the destination.

If data linkage throughout the entire system is more important than conformity

between the source database and the target databases at all destinations, specify reduced-mode transmission processing.

(b) Operation when reduced-mode transmission processing is not specified

When Datareplicator cannot swap extraction information queue files, it stops extracting update information from the system log file until space becomes available in the next extraction information queue file to be used; space will become available when the update information the file contains has finally been transmitted to its target system. Therefore, conformity is guaranteed between the source and target databases.

If it is important to guarantee conformity between the source and target databases in the overall data linkage system, do not specify reduced-mode transmission processing.

(5) Designing the unit of transmission environment definition

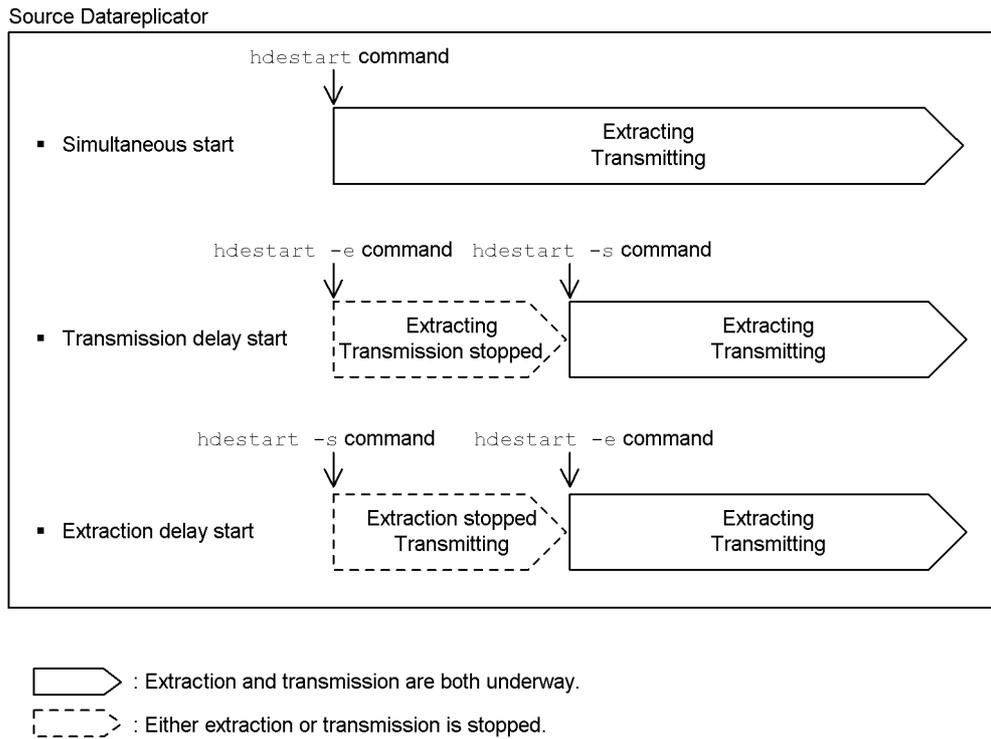
When the source HiRDB is a parallel server, the source Datareplicator requires a transmission environment definition for each back-end server. In such a case, the source Datareplicator's transmission environment definition can include the environment common to all back-end servers and the environment for a specific back-end server.

You define the common environment under `commondef` and a specific environment under `besdef (server-name)` in the transmission environment definition. If you specify the same operand in both `commondef` and `besdef (server-name)`, the specification in `besdef` is effective for the back-end server specified in `besdef`.

4.6.5 Designing the extraction processing start method

You can design the start method for extraction processing. The three extraction processing start methods are *simultaneous start*, *transmission delay start*, and *extraction delay start*. The following figure shows the relationship between the extraction processing start method and the start timing.

Figure 4-35: Relationship between the extraction processing start method and the start timing



(1) Simultaneous start

This method starts extraction and transmission when the source Datareplicator starts. When the simultaneous start method is used, update information is sent from the system log file in the order it is extracted.

- Purpose
This method is appropriate for sending update information from the system log file to the target system in the order it is extracted.
- Command execution
Execute the `hdestart` command.

(2) Transmission delay start

This method starts only extraction processing when the source Datareplicator starts and starts transmission processing when it is requested by entry of the `hdestart -s` command. The transmission delay start method is used when the target system has not started or to avoid performing extraction processing when the target system's workload

is high.

When the transmission delay start method is used, the update information stored in the extraction information queue file is not sent; instead, the extracted update information is stored. Therefore, you must allocate a sufficient space for the extraction information queue file.

- Purpose

This method is appropriate when the target system has not started or to delay extraction processing start at a time when the target system's workload is high.

Command execution

Execute the `hdestart -e` command. To start transmission processing, execute the `hdestart -s` command. By specifying a target identifier in the `hdestart -s` command, you can start transmission to a specified destination.

(3) Extraction delay start

This method starts only transmission processing when the source Datareplicator starts and starts extraction processing when it is requested by entry of the `hdestart -e` command. The extraction delay start method is used when the source HiRDB's online workload is important.

When the extraction delay start method is used, the system log file will not be placed in extraction completed status (it will not be swappable) because Datareplicator does not extract update information from the source HiRDB's system log file. If a large amount of transaction processing is expected at the source HiRDB, you must allocate a sufficient space for the system log file.

- Purpose

This method is appropriate when the source HiRDB's online workload is important.

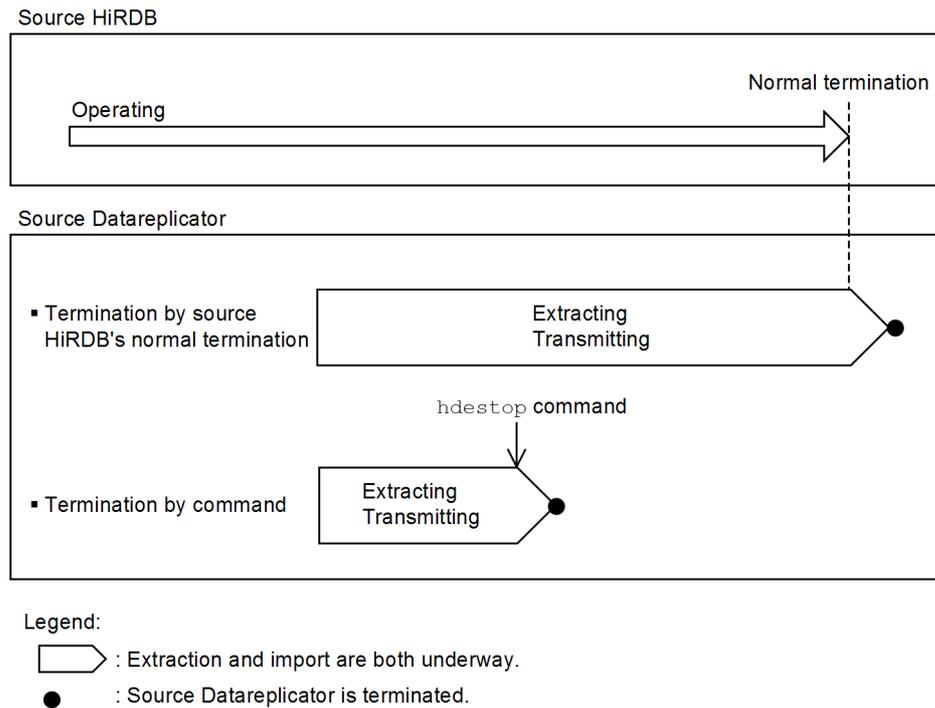
- Command execution

Execute the `hdestart -s` command. By specifying a target identifier in the `hdestart -s` command, you can start transmission to a specified destination. To start extraction processing, execute the `hdestart -e` command.

4.6.6 Designing the extraction processing stop method

You can design the stop method for extraction processing. The extraction processing stop method is simultaneous stop. The simultaneous stop method stops both extraction and transmission processing and terminates the source Datareplicator. The following figure shows the relationship between the extraction processing stop method and the stop timing.

Figure 4-36: Relationship between the extraction processing stop method and the stop timing



(1) Termination by the source HiRDB's normal termination (normal termination)

This method terminates the source Datareplicator automatically when the source HiRDB terminates normally.

- Purpose
 - This method is appropriate when you want the source Datareplicator to terminate automatically when the source HiRDB terminates normally.
- Specification of the extraction system definition
 - Specify `true` in the `syncterm` operand.
- Termination conditions
 - The source HiRDB's normal termination is detected.
 - No transactions have occurred at the source HiRDB since detection of normal termination.
 - All extraction processes are running normally and all update information has been extracted from the system log file into the extraction information queue

file.

- All transmission processes are running normally and all update information has been transmitted from the extraction information queue file to the target system.

(2) Termination by command (forced termination)

This method terminates the source Datareplicator when the extraction and transmission processing that is underway at the time the `hdestop` command is executed has terminated.

- Purpose

This method is appropriate when you want to terminate the source Datareplicator immediately.

- Command execution

Execute the `hdestop` command.

4.6.7 Designing the event control table

You use the *event facility* to manipulate the target Datareplicator's operations from the source Datareplicator.

To use the event facility, you must specify in the target environment definition for the target Datareplicator the processing that is to correspond to each event code. For details about how to specify event codes for the target Datareplicator, see [4.7.3 Designing the import procedure](#).

To use the event issuing facility, execute the `hdeevent` command at the source Datareplicator with an event code specified. The `hdeevent` command issues an SQL statement to the event control table and outputs update information for event control to the system log file. The source Datareplicator extracts this update information for event control and sends it to the target Datareplicator in order to manipulate its processing. Immediately upon detection of update information for event control, the source Datareplicator starts transmission to the target Datareplicator without waiting for the transmission interval. After the transmission processing is completed, the source Datareplicator waits for the next transmission interval.

You must create an event control table at the source HiRDB before starting the source Datareplicator.

(1) Event code specification range

The following table shows the permitted range of event codes and the corresponding actions.

Table 4-34: Event code ranges and corresponding actions

Range	Action
0	Terminates the transmission process (information is not sent to the target Datareplicator).
1 to 255	Sends the specified event code to the target Datareplicator.
Other	Information is not sent to the target Datareplicator.

(2) SQL statement issued when the hdeevent command is executed

When you execute the `hdeevent` command, the SQL statement shown below is issued to the source HiRDB. If the source HiRDB is a parallel server, the SQL statement is issued to all back-end servers.

```
LOCK TABLE "hde_dtbl" IN SHARE MODE
UPDATE "hde_dtbl" SET EVNO = user-specified-event-code
COMMIT WORK
```

Instead of executing the `hdeevent` command, you can issue a similar SQL statement directly to the target Datareplicator. However, you must not take any other action using the event control table.

(3) Conditions of the event control table

The following describes the conditions for creating an event control table before starting the source Datareplicator.

(a) Creator's user ID

Create the event control table with any user ID.

When you execute the `hdeevent` command, you will specify the user ID used to create the event control table. If you omit the user ID when you execute the `hdeevent` command, the value of the `PDUSER` environment variable will be assumed.

(b) Table name

The table name is always `hde_dtbl`.

(c) Table attribute

Specify the `FIX` attribute.

(d) Table structure

The following table shows the structure of an event control table.

Table 4-35: Structure of an event control table

Column name	Column attribute	Column length	Value
KEY	INTEGER	1	integer

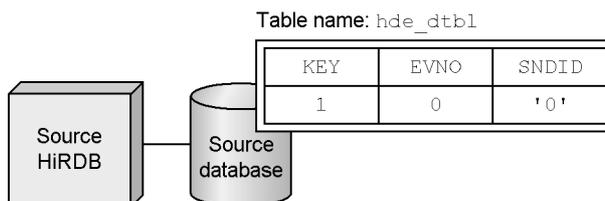
Column name	Column attribute	Column length	Value
EVNO	INTEGER	1	0
SNDID	CHAR	10	'0'

(e) Examples of event control tables

- When the source HiRDB is a single server

When the source HiRDB is a single server, you create a table consisting of one row. The following figure shows an example of an event control table when the source HiRDB is a single server.

Figure 4-37: Example of event control table when the source HiRDB is a single server



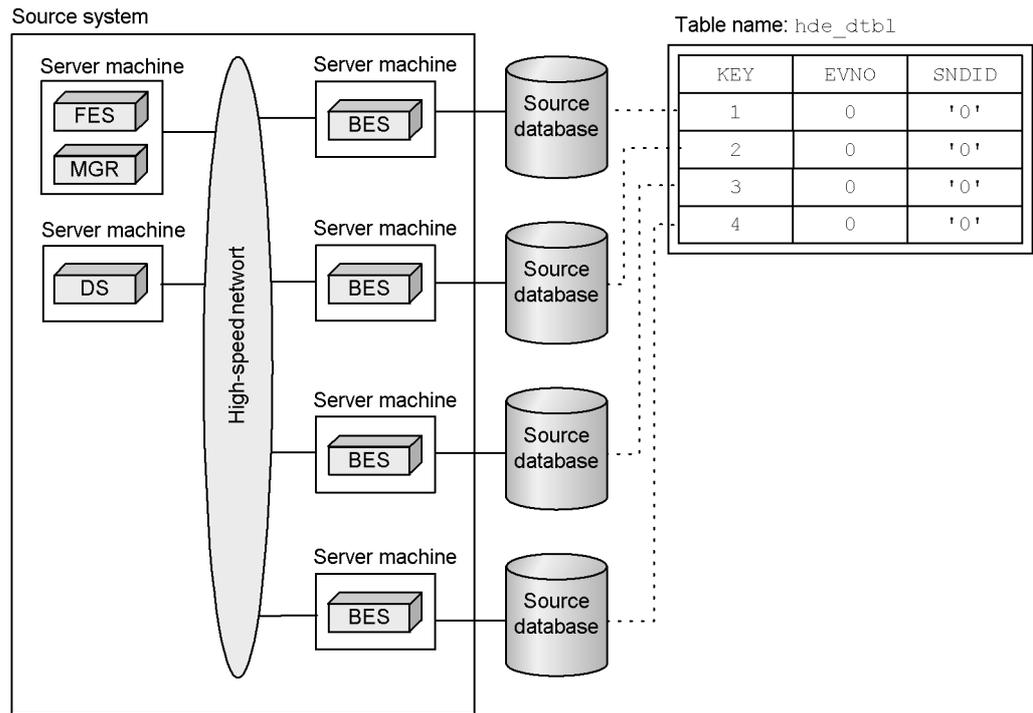
Example of SQL statement for table creation

```
CREATE FIX TABLE "hde_dtbl" ("KEY" INTEGER, "EVNO" INTEGER, "SNDID" CHAR(10))
INSERT INTO "hde_dtbl" ("KEY", "EVNO", "SNDID") VALUES (1, 0, '0')
```

- When the source HiRDB is a parallel server

When the source HiRDB is a parallel server, you create a table with one row for each key range partition at each back-end server subject to extraction processing. You must specify one RDAREA for each back-end server. The following figure shows an example of an event control table when the source HiRDB is a parallel server.

Figure 4-38: Example of event control table when the source HiRDB is a parallel server



Example of SQL statement for table creation

```
CREATE FIX TABLE "hde_dtbl" ("KEY" INTEGER, "EVNO" INTEGER, "SNDID" CHAR(10))
  IN ((RDA1)"KEY"=1, (RDA2)"KEY"=2, (RDA3)"KEY"=3, (RDA4)"KEY"=4)
INSERT INTO "hde_dtbl" ("KEY", "EVNO", "SNDID") VALUES (1, 0, '0')
INSERT INTO "hde_dtbl" ("KEY", "EVNO", "SNDID") VALUES (2, 0, '0')
INSERT INTO "hde_dtbl" ("KEY", "EVNO", "SNDID") VALUES (3, 0, '0')
INSERT INTO "hde_dtbl" ("KEY", "EVNO", "SNDID") VALUES (4, 0, '0')
```

MGR: HiRDB system manager

FES: HiRDB front-end server

BES: HiRDB back-end server

DS: HiRDB dictionary server

Note: Specify one RDAREA for each back-end server.

(f) Notes

- You must create the event control table before executing the hdeprep command.
- If the source HiRDB is a parallel server and a change is made to the configuration, such as by adding or deleting back-end servers, you must re-create the event

control table. Otherwise, no events will be issued for extractions from an added back-end server, resulting in loss of conformity between the source and target databases.

- If you have re-created the event control table, re-execute the `hdeprep` command.
- Do not create any table named `hde_dtbl` in the source HiRDB system other than the event control table.
- The event control table is not subject to extraction processing.
- If multiple events with an event code in the range 1 to 255 are issued within the same transaction, only the last event issued is effective.

4.6.8 Designing the source Datareplicator's resources

This section explains how to design the source Datareplicator's disk and memory resources.

(1) Designing the source Datareplicator's disk resources

The following table lists the source Datareplicator's disk resources.

Table 4-36: >Source Datareplicator's disk resources

File		File type		Required/optional [number of files]	See	Action when file is full
		R	C			
Definition files	Extraction system definition file	Y	N	Required [1 per source system (1 per MGR if the source HiRDB is a parallel server)]	(a)	None
	Extraction environment definition file	Y	N	Required [1 per source system (1 per MGR if the source HiRDB is a parallel server)]		
	Transmission environment definition file	Y	N	Required [1 per destination (1 per MGR for each target identifier if the source HiRDB is a parallel server)]		
	Extraction definition file	Y	N	Required [1 per source system (1 per MGR if the source HiRDB is a parallel server)]		
	Duplexing definition file	Y	N	Optional [1 per source system (1 per MGR if the source HiRDB is a parallel server)]		
Extraction definition preprocessing file		Y	Y ^{#2}	Required [1 per source system (1 per MGR if the source HiRDB is a parallel server)]	(b)	None
Extraction information queue files ^{#2}		Y	Y ^{#2}	Required [2 to 16 per source system (2 to 16 per BES if the source HiRDB is a parallel server)]	(c)	Swapped ^{#3}
Status files	Extraction master status file	Y	Y ^{#2}	Required [1 per source system (1 per MGR if the source HiRDB is a parallel server)]	(d)	None
	Extraction server status file ^{#2}	Y	Y ^{#2}	Required [1 per source system (1 per BES if the source HiRDB is a parallel server)]		

File		File type		Required/optional [number of files]	See	Action when file is full
		R	C			
Error information files	Extraction master error information files	Y	N	Required [2 per source system (2 per MGR if the source HiRDB is a parallel server)]	(e)	Swapped
	Extraction node master error information files	Y	N	Required [2 per source system (2 per server machine containing a BES if the source HiRDB is a parallel server)]		
Activity trace files	Extraction master trace files	Y	N	Optional [2 per source system]	(f)	Swapped
	Extraction node master trace files	Y	N	Optional [2 per source system (2 per BES if the source HiRDB is a parallel server)]		
Data linkage file		Y	Y ^{#2}	Required [1 per source system (1 per BES if the source HiRDB is a parallel server)]	(g)	None
Command log files		Y	N	Optional [2 per source system]	(h)	Swapped

Legend:

BES: Back-end server (includes back-end servers that do not contain any databases subject to extraction processing)

R: UNIX regular file or Windows file

C: UNIX character special file

Y: Can be created.

N: Cannot be created.

#1

If transmission processing is not completed on the file to be used after swapping, the source Datareplicator waits for the next transmission interval. If transmission processing is completed by then, the source Datareplicator starts extracting the update information. For details about the transmission interval, see 4.6.4 *Designing the transmission procedure*.

#2

If you use character special files with the AIX edition, add 1,024 bytes to the formula for determining the size of each file. In the `queuesize` operand in the extraction environment definition, specify the size obtained from the formula minus 1,024 bytes.

#3

If you use a Datareplicator file system area, also see 3.5.3(1) *Rules for allocating a Datareplicator file system area*.

(a) Sizes of definition files

The sizes of the source Datareplicator's definition files (extraction system definition file, extraction environment definition file, transmission environment definition file, and extraction definition file) depend on the definition operands that are specified.

(b) Size of the extraction definition preprocessing file

The source Datareplicator converts the specified extraction definition to the interval format, and then stores it in the extraction definition preprocessing file. The following are the formulas for determining the size of the extraction definition preprocessing file:

- For a regular file for UNIX or a Windows file
 $164 + (40 + SV_INFO) \times SV_NUM$ (bytes)
- For a character special file for UNIX
 $\uparrow 164 / SCT_SIZE \uparrow \times SCT_SIZE + (40 + SV_INFO) \times SV_NUM$ (bytes)
- *SV_INFO*
 $= 44 + (52 + 96 \times SCH_NUM + 64 \times UPD_NUM + 128 \times COL_NUM + 64 + 112 \times TYPE_NUM + 64 \times ATTR_NUM) + 24 + 32 \times TBL_NUM + 16 \times COL_NUM + SND_INFO + LOB_INFO$
- *SND_INFO*
 $= 16 + 28 \times UPD_NUM + \sum_{SND} (28 \times SND_UPD + 32 \times SND_COND + 272 \times SND_CNST) + 16 + 40 \times SND_NUM + \sum_{SND} (12 \times SND_UPD + 32 \times SND_TBL + 16 \times SND_COL)$
- *LOB_INFO*
 $= 16 + 4 \times LOB_AREA$

The following explains the variables used in the above formulas:

- \sum_{SND} : Total sum of the number of destinations
- *SV_NUM*: Number of back-end servers

- *SV_INFO*: Size of definition information per back-end server
- *SCH_NUM*: Total number of schemas subject to extraction
- *UPD_NUM*: Total number of update information names
- *TBL_NUM*: Total number of tables subject to extraction
- *COL_NUM*: Total number of columns subject to extraction
- *TYPE_NUM*: Number of abstract data types subject to extraction, excluding duplications (this value includes the number of high-order data types inherited by abstract data types subject to extraction and the number of abstract data types defined as an abstract data type's attribute data type)
- *ATTR_NUM*: Total number of abstract data type attributes included in *TYPE_NUM*
- *SND_INFO*: Size of destination information
- *SND_NUM*: Number of destinations
- *SND_UPD*: Number of update information names subject to transmission per destination
- *SND_TBL*: Number of tables subject to transmission per destination
- *SND_COL*: Number of columns subject to transmission per destination
- *SCT_SIZE*: Sector size
- *SND_COND*: Number of transmission conditions per destination
- *SND_CNST*: Number of transmission condition constants specified per destination
- *LOB_INFO*: Size of BLOB extraction information
- *LOB_AREA*: Number of BLOB column storage RDAREAs subject to extraction

(c) Size of an extraction information queue file

Extracted update information is accumulated in an extraction information queue file. The size of an extraction information queue file must be greater than the size of the update information to be extracted.

The following is the formula for determining the size of an extraction information queue file (formula for calculating the size of update information); the formula is the same whether the file type is OS regular file or character special file:

Formula for calculating the size of update information

$EXT_ALL + QFL_NUM \times SCT_SIZE + (EXT_ALL / IO_SIZE) \times 12$ (bytes)

- *EXT_ALL*

$$= \Sigma_{ins}(INS_SIZE) + \Sigma_{upd}(UPD_SIZE) + \Sigma_{del}(DEL_SIZE) + 168 \times TRN_NUM + 122 \times EVT_NUM$$

- *INS_SIZE*
= 100 + *EXT_COL* × 12 + *ROW_SIZE*
- *UPD_SIZE*
= 100 + *EXT_COL* × 12 + *ROW_SIZE*
- *DEL_SIZE*
= 100 + *MAP_COL* × 12 + $\Sigma_{map}(MAP_DATA)$
- *ROW_SIZE*
= $\Sigma_{ext}(COL_DATA + LOB_DATA + VAR_DATA + ADT_DATA + MLT_DATA)$
- *COL_DATA*
= $\lceil \text{column-data-length} / 4 \rceil \times 4$
- *LOB_DATA*
= 56 × $\lceil (BLOB\text{-data-length} / 8,192) \rceil + BLOB\text{-data-length}$
- *VAR_DATA*
= 48 × $\lceil (VARCHAR\text{-data-length} / PAGE_SIZE) \rceil + VARCHAR\text{-data-length}$
- *ADT_DATA*
= 48 × *ADT_NUM* + *abstract-data-type-data-length*
For details about the data length of an abstract data type, see the applicable HiRDB manual.
- *MLT_DATA*
= 58 × $\lceil \lceil \text{maximum-elements-count} / 8 \rceil / 4 \rceil \times 4 + 52 \times \lceil \text{repetition-column-data-length} / PAGE_SIZE \rceil + \text{repetition-column-data-length}$
repetition-column-data-length:
 - Fixed-length element: 1 + *defined-element-length* × *elements-count*
 - Variable-length element: $\Sigma_{elm}(5 + VAR_DATA)$

The following explains the variables used in the above formulas:

- *EXT_ALL*: Total size of update information
- *SCT_SIZE*: Sector size

- *IO_SIZE*: Value of `quiosize` operand in the extraction environment definition
- *INS_SIZE*: Size of unit insertion row
- *UPD_SIZE*: Size of unit update row
- *DEL_SIZE*: Size of unit deletion row
- *ROW_SIZE*: Length of update data per row
- *QFL_NUM*: Number of extraction information queue files
- *TRN_NUM*: Number of transactions executed for HiRDB subject to extraction (including transactions on a table that is not subject to extraction, and invalid transaction rollbacks)
- *EVT_NUM*: Number of events issued
- *EXT_COL*: Number of columns subject to extraction per row
- *MAP_COL*: Number of mapping key columns per row
- *LOB_DATA*: Length of update data in BLOB column
- *VAR_DATA*: Length of update data for a VARCHAR (including MVARCHAR and NVARCHAR) column with a length of 256 bytes or greater
- *PAGE_SIZE*: Page size of HiRDB database
- *ADT_DATA*: Length of update data in an abstract data type column
- *ADT_NUM*: Number of abstract data types (including the associated columns of inherited data type and nested data type)
- *MLT_DATA*: Length of update data in a repetition column
- *COL_DATA*: Length of update data per column (excluding BLOB columns, VARCHAR columns with a length of 256 bytes or greater, and columns of abstract data type)
- *MAP_DATA*: Size of mapping key
- Σ_{ins} : Total sum of insertion row sizes
- Σ_{upd} : Total sum of update row sizes
- Σ_{del} : Total sum of deletion row sizes
- Σ_{map} : Total sum of mapping key sizes per row
- Σ_{ext} : Total sum of column data sizes subject to extraction per row
- Σ_{elm} : Total sum of repetition columns' element sizes per row

Notes on estimating the size of an extraction information queue file

Note the following about estimating the size of an extraction information queue file:

- Provide an extraction information queue file that is large enough to store three to four days' worth of update information, taking into account the time required for recovering the target system in the event of a system shutdown.
- If transactions such as those listed below occur and the extraction information queue file becomes full, the contents of the source and target databases must be synchronized, and then Datareplicator must be initialized:
 - A batch transaction was executed and the update information generated within one transaction cannot be stored in the extraction information queue file.
 - A transaction that updated the source database was left in uncompleted status and not all of the update information generated by multiple other transactions can be stored in the extraction information queue file.
- Differences in size depending on the extraction handling method

The table below explains the differences in the size estimation for the different extraction handling methods. Note that the size of an extraction information queue file must be at least the maximum size of a single transaction (maximum size of data that can be updated).

If a single transaction cannot fit in the extraction information queue file, extraction processing cannot continue. If HiRDB's system log file becomes full due to accumulated unextracted logs, the HiRDB server will shut down.

Table 4-37: Notes on the size estimation depending on the extraction handling method

Handling method	Notes
Simultaneous start or extraction delay start	<ul style="list-style-type: none"> • Take into account the delay in extraction at the source Datareplicator and use the amount of data that can be accumulated at the peak of transactions as the size of the disk space. • Operation is possible with just two files. However, if transmission processing alone might be stopped for some reason, we recommend that you increase the number of files.
Transmission delay start	<ul style="list-style-type: none"> • Calculate the size of the disk space by taking into account the amount of data that can be accumulated before and after startup of transmission processing. • More than two files might be required, depending on the amount of data accumulated, because the maximum size of an extraction information queue file is 2GB. However, if you are using both simultaneous start and transmission delay start, it might be easier to provide more smaller files to avoid a shortage of file space.

(d) Sizes of status files

The following are the formulas for determining the sizes of the status files (extraction master status file and extraction server status file) for a source Datareplicator.

For status files, the required size is allocated when the source Datareplicator is initialized. Status files will never become full because the file capacity is increased during the actual replication.

Formulas for determining the size of the extraction master status file

- For a regular file for UNIX or a Windows file
 $(2 + SND_NUM + ND_NUM \times SV_NUM) \times 1,024$ (bytes)
 - For a character special file for UNIX whose sector size exceeds 1 kilobyte
 $(2 + SND_NUM) + (ND_NUM \times SV_NUM) \times SCT_SIZE$ (bytes)
- ND_NUM*: Total number of nodes
SV_NUM: Total number of servers
SND_NUM: Total number of destinations
SCT_SIZE: Sector size

Formulas for determining the size of the extraction server status file

If the source HiRDB is a parallel server, determine the size of the status file for each back-end server.

- For UNIX
 $(22 + SND_NUM \times 2 + UINF_NUM + SYS_NUM) \times 1,024$ (bytes)
 - For Windows
 $(22 + SND_NUM \times 2 + UINF_NUM) \times 1,024$ (bytes)
 - For a character special file for UNIX whose sector size exceeds 1 kilobyte
 $(22 + SND_NUM \times 2 + UINF_NUM + SYS_NUM) \times SCT_SIZE$ (bytes)
- SND_NUM*: Number of destinations
UINF_NUM: Value of the `extinforonum` operand in the extraction system definition
SCT_SIZE: Sector size
SYS_NUM: Number of file system areas specified in the extraction environment definition

(e) Sizes of the error information files

The default size of an error information file (extraction master error information file

or extraction node master error information file) for the source Datareplicator is 16KB. To retain error information for a long period of time, increase the size of the file by changing the extraction system definition.

(f) Sizes of activity trace files

To specify the sizes of activity trace files (extraction master trace files and extraction node master trace files), use the `int_trc_filesz` operand in the extraction system definition.

(g) Size of the data linkage file

The following are the formulas for determining the size of the data linkage file; if the source HiRDB is a parallel server, determine a data linkage file size for each back-end server:

- For a regular file for UNIX or a Windows file
9 x 1,024 (bytes)
- For a character special file for UNIX whose sector size exceeds 1 kilobyte
9 x *SCT_SIZE* (bytes)
SCT_SIZE: Sector size

(h) Size of a command log file

The size is always 128 KB.

(i) Sizes of other files

The Windows edition of Datareplicator creates several work files in the `tmp` directory under the installation directory. Therefore, provide 4 MB of space for the work files.

(2) Designing the source Datareplicator's memory resources

The following table lists the memory resources for the source Datareplicator.

Table 4-38: List of memory resources for the source Datareplicator

Memory resource	Subsection
Extraction master process	(a)
Extraction node master process	(b)
Extraction process	(c)
Transmission process	(d)
Transmission master process	(e)
Activity trace collection process	(f)

Memory resource	Subsection
Datareplicator agent process	(g)
Size of shared memory for command communication	(h)
Size of shared memory for reporting status information	(i)
Size of shared memory for process-to-process communication	(j)
Size of shared memory for storing extraction definition	(k)
Size of shared memory for storing messages	(l)
Size of shared memory for storing transaction management information	(m)
Number of required semaphores	(n)
Number of required message queues	(o)
Size of update information editing buffer	(p)

Table 4-39: List of variables used in the formulas for determining the size of memory for source Datareplicator

Variable name	Description of variable
<i>ATTR_NUM</i>	Total number of abstract data type attributes subject to data linkage
<i>COL_LEN</i>	Maximum value of the total lengths of columns subject to extraction in a table subject to extraction. For the columns listed below, use the indicated length and obtain the total length: <ul style="list-style-type: none"> • Repetition columns: Definition length for one element • BLOB and BINARY columns: <ul style="list-style-type: none"> If the length is 35,000 bytes or less: Column definition length If the length is greater than 35,000 bytes: 35,000 bytes • ADT: 0
<i>COL_NUM</i>	Total number of columns in all tables subject to extraction
<i>COM_FILE</i>	Size of data linkage file (bytes)
<i>DATA_NUM</i>	Maximum number of update information items that can occur within the transmission interval. If there is a transaction whose processing time is longer than the transmission interval (such as a batch transaction), the update information items are accumulated until the transaction is completed.
<i>EMST_FILE</i>	Size of extraction master status file (bytes)
<i>EQUE_FILE</i>	Size of extraction information queue file (bytes)

4. System Design

Variable name	Description of variable
<i>EST_FILE</i>	Size of extraction server status file (bytes)
<i>EXT_FILE</i>	Size of extraction environment definition file (bytes)
<i>LOB_MAX</i>	Maximum length of update data (bytes) in all BLOB attribute columns subject to extraction (actual data length, not the column definition length)
<i>LOGIOSIZE</i>	logiosize value specified in the extraction environment definition (specify this value in KB)
<i>MSV_NUM</i>	When the source is HiRDB: Maximum number of back-end servers in each node subject to extraction When the source is not HiRDB: 1
<i>ND_NUM</i>	Total number of nodes subject to extraction
<i>NSV_NUM</i>	Number of servers in the corresponding node
<i>PRP_FILE</i>	Size of extraction definition preprocessing file (bytes)
<i>QUE_NUM</i>	qufileXXX value specified in the extraction environment definition
<i>QUIOSIZE</i>	quiosize value specified in the extraction environment definition (specify this value in KB)
<i>RBUF_NUM</i>	readbufnum value specified in the transmission environment definition
<i>RDAMAX_ALL</i>	Maximum page length of RDAREA that stores tables subject to linkage (bytes)
<i>RDAMAX_BIN</i>	Maximum page length of RDAREA that stores BINARY data (bytes)
<i>RDAMAX_LOB</i>	Maximum page length of RDAREA that stores LOB data (bytes)
<i>RDAMAX_REP</i>	Maximum page length of RDAREA that stores tables containing repetition columns (bytes)
<i>RDAMAX_VCHR</i>	Maximum page length of RDAREA that stores tables containing varchar, nvarchar, or mvarchar type (bytes)
<i>SCH_NUM</i>	Total number of authorization identifiers subject to extraction
<i>SND_FILE</i>	Size of transmission environment definition file (bytes)
<i>SND_NUM</i>	Number of sendidx operands specified in the extraction system definition

Variable name	Description of variable
<i>SV_NUM</i>	When the source is HiRDB: Total number of back-end servers subject to extraction When the source is not HiRDB: 1
<i>SYS_FILE</i>	Size of extraction system definition file (bytes)
<i>TBL_NUM</i>	Total number of tables subject to extraction
<i>TRN_NUM</i>	Maximum number of transactions detected during the transmission interval. If there are transactions whose processing time is longer than the transmission interval (such as batch transactions), the number of transactions is accumulated until such transactions are completed.
<i>TYPE_NUM</i>	Total number of abstract data types in all tables subject to extraction
<i>UBUF</i>	When the source database is HiRDB: <ul style="list-style-type: none"> If <i>nodemst</i> is specified in <i>sendcontrol</i> in the extraction system definition <i>editbufsize</i> value specified in the transmission environment definition If <i>sendmst</i> is specified in <i>sendcontrol</i> in the extraction system definition <i>smt_editbufsize</i> value specified in the extraction system definition When the source database is XDM/DS: <i>REFLECTBUFF</i> value specified in the XDM/DS start definition. For details, see the manual <i>VOS3 XDM Data Linkage Facility XDM/DS Description and Definition</i> . (Specify this value in KB)
<i>UINF_NUM</i>	<i>extinfunum</i> value specified in the extraction system definition

(a) Extraction master process

Number of processes

If the source Datareplicator is a single server, there is only one process. If the source Datareplicator is a parallel server, there is one process per server machine that contains a system manager.

Size of procedure

163,840 bytes

Dynamic memory size (bytes)

The dynamic memory size is the sum of the fixed part shown below and any applicable variable part explained below.

Fixed part:

$$\begin{aligned}
 & (\downarrow ND_NUM / 8 \downarrow + 1) \times 8,800 \\
 & + (\downarrow SV_NUM / 16 \downarrow + 1) \times 17,600 \\
 & + PRP_FILE \times 5 \\
 & + MSV_NUM \times (39,196 + SND_NUM \times 3,968 + UINF_NUM \times 1,024) \\
 & + (76 + 32 \times SND_NUM) \times MSV_NUM \\
 & + (8 + SND_NUM + 18 \times SV_NUM) \times 132 \\
 & + MAX(SYS_FILE, EXT_FILE, SND_FILE) \\
 & + 58,658
 \end{aligned}$$

Variable part:

- Formula to be added if the import transaction synchronization facility is used
 - $48 \times UINF_NUM$
 - $+ 10 \times SV_NUM$
 - $+ 1,024 \times SV_NUM$
 - $+ 1,494$
- Formula to be added if the file duplexing function is used
 - $1,024$

Explanation of the variables

See Table 4-39 *List of variables used in the formulas for determining the size of memory for source Datareplicator.*

(b) Extraction node master process

Number of processes

If the source Datareplicator is a single server, there is only one process. If the source Datareplicator is a parallel server, there is one process per server machine that contains a back-end server.

Size of procedure

71,680 bytes

Dynamic memory size (bytes)

The dynamic memory size is the sum of the fixed part shown below plus the variable part explained below if applicable.

Fixed part:

$$\begin{aligned}
& (100 \times NSV_NUM + 744 \times SND_NUM \times NSV_NUM) \\
& + (32 \times SND_NUM \times NSV_NUM) \\
& + (2 \times SND_NUM) \\
& + QUIOSIZE \times 1,024 \\
& + (224 + 2,040 + MSV_NUM \times (39,196 + SND_NUM \times 3,968 + UINF_NUM \\
& \times 1,024) + PRP_FILE) \\
& + (\text{MAX}(QUIOSIZE \times 1,024 + 516, 66,052)) \\
& + (\downarrow QUIOSIZE \times 1,024 / 1,024 \downarrow \times 1,024) \\
& + (\text{MAX}(EMST_FILE, EST_FILE, EQU_FILE, COM_FILE) - 1,024) \\
& + 545,086
\end{aligned}$$

Variable part:

- Formula to be added if the file duplexing function is used
(QUIOSIZE \times 1,024)

Explanation of the variables

See Table 4-39 *List of variables used in the formulas for determining the size of memory for source Datareplicator.*

(c) Extraction process

Number of processes

If the source Datareplicator is a single server, there is only one process. If the source Datareplicator is a parallel server, there is one process per server machine that contains a back-end server.

Size of procedure

153,600 bytes

Dynamic memory size (bytes)

The dynamic memory size is the sum of the fixed part explained below plus the variable part shown below if applicable.

Fixed part:

$$\begin{aligned}
& (1,024 \times (UINF_NUM + 1)) \\
& + (2,944 \times SND_NUM) \\
& + LOGIOSIZE \times 1,024 \\
& + (\uparrow\uparrow SND_NUM / 8 \uparrow / 4 \uparrow \times 4)
\end{aligned}$$

$$\begin{aligned}
&+ (12 \times QUE_NUM) \\
&+ (LOGIOSIZE \times 2,048) \\
&+ (MAX((4,184 + \uparrow\uparrow SND_NUM / 8 \uparrow / 4 \uparrow \times 4 \\
&+ 15 \times COL_NUM + COL_LEN + 4,096), (LOGIOSIZE \times 2,048))) \\
&+ (MAX(QUIOSIZE \times 1,024 + 516, 66,052) + (QUIOSIZE \times 1,024)) \\
&+ (\downarrow QUIOSIZE \times 1,024 / 1,024 \downarrow \times 1,024) \\
&+ (MAX(EMST_FILE, EST_FILE, EQUE_FILE, COM_FILE) - 1,024) \\
&+ (LOGIOSIZE \times 1,024) \\
&+ 744,992
\end{aligned}$$

Variable part:

- Formula to be added if the file duplexing function is used
 $QUIOSIZE \times 1,024$

Explanation of the variables

See Table 4-39 *List of variables used in the formulas for determining the size of memory for source Datareplicator.*

(d) Transmission process

When nodemst is specified in the sendcontrol operand in the extraction system definition

- Number of processes
 If the source Datareplicator is a single server, there is only one process. If the source Datareplicator is a parallel server, there is one process per server machine that contains a back-end server.
- Size of procedure
 378,880 bytes
- Dynamic memory size (bytes)
 The dynamic memory size is the sum of the fixed part shown below plus any applicable variable part explained below.

Fixed part:

$$\begin{aligned}
&(UBUF \times 1,024 - 120) \\
&+ (12 \times QUE_NUM) \\
&+ MAX(88 + \uparrow SND_NUM / 32 \uparrow \times 4, 1,244)
\end{aligned}$$

$$\begin{aligned}
&+ (QUIOSIZE \times 1,024) \\
&+ ((TRN_NUM \times 340 + DATA_NUM \times 64)) \dots * \\
&+ \uparrow(SND_NUM / 32) \uparrow \times 4 \\
&+ UBUF \times 1,024 \\
&+ 32 \times TBL_NUM \\
&+ 96 \times SCH_NUM + 64 \times TBL_NUM + 128 \times COL_NUM \\
&+ \text{MAX}(QUIOSIZE \times 1,024 + 516, 66,052) + QUIOSIZE \times 1,024 \\
&+ (\downarrow QUIOSIZE \times 1,024 / 1,024 \downarrow \times 1,024) \times RBUF_NUM \\
&+ 18,604
\end{aligned}$$
Important:

For the portion of the formula indicated by the asterisk (*), if there are more transactions or update information than estimated, the transmission process will extend the memory automatically. Because there is no limit to the number of such memory extensions, the size of a transmission process might become very large if a large volume of update processing is performed on the source database by a single transaction, such during batch processing. If you apply data linkage to batch processing, perform commit processing periodically so that no more than a few tens of thousands of items are updated by a single transaction.

In AIX, if the estimated size exceeds 256 megabytes, set the LDR_CNTRL environment variable. For details about how to set the environment variable, see *2.4.1 Environment variables for a source Datareplicator* and *2.5.1 Setting up a source Datareplicator's communications environment*.

Variable part:

- Formula to be added if the columns subject to extraction include the following data types
 - varchar, nvarchar, mvarchar, blob, binary, repetition column
Number of branch data items x 108
Note: Guideline for branch data: Total data length in variable-length columns ÷ average value for RDAREA
 - If you use the varchar, nvarchar, mvarchar, or BLOB type, also add the following:
 $\text{MAX}(RDAMAX_VCHR, 8,192)$

- If you use the BLOB type, also add the following:
 $1,024 \times \uparrow \uparrow LOB_MAX / RDAMAX_LOB \uparrow / 16 \uparrow$
- If you use repetition columns, also add the following:
 $RDAMAX_REP \times 2$
- If you use the binary type, also add the following:
 $RDAMAX_BIN$
- If you use the SGML type, also add the following:
 $64 + 112 \times TYPE_NUM + 64 \times ATTR_NUM$
- Formula to be added if the import transaction synchronization facility is used
 $32 + SV_NUM$
- Formula to be added if a transmission UOC routine or the SGML type is used
 $(68 \times TBL_NUM + 16 \times COL_NUM) + (64 \times COL_NUM)$
- Formula to be added if the HDE_BIN_COL_MAXLEN environment variable is specified or a transmission UOC routine is used
 $20 \times TBL_NUM + 8 \times COL_NUM$
- Formula to be added if ukey is specified in the extraction definition
 $RDAMAX_ALL$
- Formula to be added if a character set is used
 $32,032$
- Formula to be added if a file system area is used or a character special file is used in AIX
 $4,608$
- Formula to be added if the file duplexing function is used
 $29,736 + (QUIOSIZE \times 1,024)$

Explanation of the variables

See Table 4-39 *List of variables used in the formulas for determining the size of memory for source Datareplicator.*

When sendmst is specified in the sendcontrol operand in the extraction system definition

- Number of processes

As many processes are required as are specified in the `sendprocnum` operand in the extraction system definition.

- Size of procedure

307,200 bytes

- Dynamic memory size (bytes)

The dynamic memory size is the sum of the fixed part shown below plus any applicable variable part explained below.

Fixed part:

$$\begin{aligned}
 & (UBUF \times 1,024 - 120) \\
 & + (12 \times QUE_NUM) \\
 & + \text{MAX}(88 + \uparrow SND_NUM / 32 \uparrow \times 4, 1,244) \\
 & + (QUIOSIZE \times 1,024) \\
 & + UBUF \times 1,024 \\
 & + 40 + 32 \times TBL_NUM \\
 & + 52 + 96 \times SCH_NUM + 64 \times TBL_NUM + 128 \times COL_NUM \\
 & + 2,428 + \text{MAX}(QUIOSIZE \times 1,024 + 516, 66,052) \\
 & + QUIOSIZE \times 1,024 + 516 \\
 & + 11,808 + (\downarrow QUIOSIZE \times 1,024 / 1,024 \downarrow \times 1,024) \times RBUF_NUM \\
 & + 18,588
 \end{aligned}$$

Variable part:

- Formula to be added if the columns subject to extraction include the following data types
 - `varchar`, `nvarchar`, `mvarchar`, `blob`, `binary`, `repetition column`

Number of branch data items \times 108

Note: Guideline for branch data: Total data length in variable-length columns \div average value for RDAREA

- If you use the `varchar`, `nvarchar`, `mvarchar`, or `BLOB` type, also add the following:

$$\text{MAX}(RDAMAX_VCHR, 8,192)$$

- If you use the BLOB type, also add the following:
 $1,024 \times \uparrow\uparrow LOB_MAX / RDAMAX_LOB \uparrow / 16 \uparrow$
- If you use repetition columns, also add the following:
 $RDAMAX_REP \times 2$
- If you use the binary type, also add the following:
 $RDAMAX_BIN$
- If you use the SGML type, also add the following:
 $64 + 112 \times TYPE_NUM + 64 \times ATTR_NUM$
- Formula to be added if the import transaction synchronization facility is used
 $32 + SV_NUM$
- Formula to be added if a transmission UOC routine or the SGML type is used
 $(68 \times TBL_NUM + 16 \times COL_NUM) + (64 \times COL_NUM)$
- Formula to be added if the HDE_BIN_COL_MAXLEN environment variable is specified or a transmission UOC routine is used
 $20 \times TBL_NUM + 8 \times COL_NUM$
- Formula to be added if ukey is specified in the extraction definition
 $RDAMAX_ALL$
- Formula to be added if a character set is used
 $32,032$
- Formula to be added if a file system area is used or a character special file is used in AIX
 $4,608$
- Formula to be added if the file duplexing function is used
 $29,736 + (QUIOSIZE \times 1,024)$

Explanation of the variables

See Table 4-39 *List of variables used in the formulas for determining the size of memory for source Datareplicator.*

(e) Transmission master process

Number of processes

If the source Datareplicator is a single server, there is only one process. If the source Datareplicator is a parallel server, there is one process per server machine that contains a back-end server.

As many processes are required as are specified in the `sendprocnum` operand in the extraction system definition.

Size of procedure

256,000 bytes

Dynamic memory size (bytes)

The dynamic memory size is the sum of the fixed part shown below plus any applicable variable part explained below.

Fixed part:

$$\begin{aligned}
 & (12 \times QUE_NUM) \\
 & + \text{MAX}(88 + \uparrow SND_NUM / 32 \uparrow \times 4, 1,244) \\
 & + (QUIOSIZE \times 1,024) \\
 & + ((TRN_NUM \times 340 + DATA_NUM \times 64) \times SND_NUM) \\
 & + \uparrow (SND_NUM / 32) \uparrow \times 4 \\
 & + \text{MAX}(QUIOSIZE \times 1,024 + 516, 66,052) + QUIOSIZE \times 1,024 \\
 & + \downarrow QUIOSIZE \times 1,024 / 1,024 \downarrow \times 1,024 \\
 & + 18,512
 \end{aligned}$$

Variable part:

- Formula to be added if a file system area is used or a character special file is used in AIX
4,608
- Formula to be added if the file duplexing function is used
 $11,736 + (QUIOSIZE \times 1,024)$

Explanation of the variables

See Table 4-39 *List of variables used in the formulas for determining the size of memory for source Datareplicator.*

(f) Activity trace collection process

Number of processes

As many processes are required as there are extraction master processes + extraction node master processes.

Size of procedure

61,440 bytes

Dynamic memory size

51,200 bytes

(g) Datareplicator agent process

Number of processes

Add the number of processes either at the source system or at the target system.

Size of procedure

71,680 bytes

Dynamic memory size (bytes)

$$300,000^{\#} + (ND_NUM + SV_NUM + SND_NUM + 2) \times 64$$

Explanation of the variables

See Table 4-39 *List of variables used in the formulas for determining the size of memory for source Datareplicator.*

#

Use the value at either the source system or the target system.

(h) Determining the size of shared memory for command communication

The following is the formula for determining the size of shared memory for command communication; use this shared memory at the server machine where the extraction master process is run:

$$\text{Size of shared memory for command communication} = 1,360 \text{ (bytes)}$$

(i) Determining the size of shared memory for reporting status information

The following is the formula for determining the size of shared memory for reporting status information; use this shared memory at the server machine where the extraction master process is run:

$$1,360 + 36 + 64 + ND_NUM + 60 + SV_NUM + 164 \times SV_NUM \times SND_NUM + 8 \times \sum_{SND} (NSND_NUM) \times SV_NUM + (ND_NUM + SV_NUM + SV_NUM \times SND_NUM) \times 4 \text{ (bytes)}$$

\sum_{SND} : Transmissions to all destinations

ND_NUM : Total number of nodes

SV_NUM : Total number of servers

SND_NUM : Total number of destinations

NSND_NUM: Number of transmission-suppressed original receiver identifiers per destination

(j) Determining the size of shared memory for process-to-process communication

The following is the formula for determining the size of shared memory for process-to-process communication; use this shared memory at the server machine where the extraction node master process is run:

$$1,048 + (2,969 + 4 \times SND_NUM + 16 \times UINF_NUM) \times NSV_NUM + 1,600 \times NSV_NUM + SND_NUM \text{ (bytes)}$$

SND_NUM: Total number of destinations

NSV_NUM: Number of servers in the corresponding node

UINF_NUM: Value of the `extinforum` operand in the extraction system definition

(k) Determining the size of shared memory for storing extraction definition

The following is the formula for determining the size of shared memory for storing extraction definition; use this shared memory at the server machine where the extraction node master process is run:

$$164 + (40 + SV_INFO) \times SV_NUM \text{ (bytes)}$$

- *SV_INFO*

$$= 44 + (52 + 96 \times SCH_NUM + 64 \times UPD_NUM + 128 \times COL_NUM + 64 + 112 \times TYPE_NUM + 64 \times ATTR_NUM) + 24 + 32 \times TBL_NUM + 16 \times COL_NUM + SND_INFO + LOB_INFO$$

- *SND_INFO*

$$= 16 + 28 \times UPD_NUM + 40 \times SND_NUM + \Sigma_{SND} (40 \times SND_UPD + 20 \times SND_TBL + 16 \times SND_COL)$$

- *LOB_INFO*

$$= 16 + 4 \times LOB_AREA$$

Explanation of the variables

Σ_{SND} : Total sum of the number of destinations

SV_NUM: Number of back-end servers in the corresponding node

SV_INFO: Size of definition information per back-end server

SCH_NUM: Total number of schemas subject to extraction

UPD_NUM: Total number of update information names

TBL_NUM: Total number of tables subject to extraction

COL_NUM: Total number of columns subject to extraction

TYPE_NUM: Number of abstract data types subject to extraction, excluding duplications. This value includes the number of high-order data types inherited by the abstract data types subject to extraction and the number of abstract data types defined as an abstract data type's attribute data type.

ATTR_NUM: Total number of abstract data type attributes included in *TYPE_NUM*

SND_INFO: Size of destination information

SND_NUM: Number of destinations

SND_UPD: Number of update information names subject to transmission per destination

SND_TBL: Number of tables subject to transmission per destination

SND_COL: Number of columns subject to transmission per destination

LOB_INFO: Size of BLOB extraction information

LOB_AREA: Number of BLOB column storage RDAREAs subject to extraction

(l) Determining the size of shared memory for storing messages

The following is the formula for determining the size of shared memory for storing messages; use this shared memory at the server machine where the extraction node master process is run:

Size of shared memory for storing messages = 79,200 (bytes)

(m) Determining the size of shared memory for storing transaction management information

The following is the formula for determining the size of shared memory for storing transaction management information; use this shared memory at the server machine where the extraction node master process is run (this shared memory is required only when *sendmst* is specified in the *sendcontrol* operand in the extraction system definition):

$1,048,576 \times NSV_NUM$ (bytes)

NSV_NUM: Number of servers in the corresponding node

(n) Determining the number of required semaphores

The following are the formulas for determining the number of semaphores that are used by the source Datareplicator; use the semaphores at the server machine where the extraction node master process is run:

When *sendcontrol=nodemst* is specified in the extraction system definition

$$1 + (2 + 2 \times SND_NUM) \times NSV_NUM$$

When `sendcontrol=sendmst` is specified in the extraction system definition

$$1 + 4 \times NSV_NUM$$

SND_NUM: Total number of destinations

NSV_NUM: Number of servers in the corresponding node

Change the maximum number of semaphores in the entire system that is specified in the kernel parameter (`SEMMNS`) so that the number of semaphores determined by the above formula can be used at the source system. For details about how to update the kernel parameter, see the applicable OS documentation.

(o) Determining the number of required message queues

One message queue is required for the server machine at which the extraction master process is run, and another message queue is required for each server machine at which the extraction node master process is run.

(p) How to estimate the size of the update information editing buffer

The formula for estimating the size of the update information editing buffer is shown below. First, estimate the size of the update information editing buffer for each table subject to extraction, and then specify the largest value in the `smt_editbufsize` operand described in 5.2 *Extraction system definition* or the `editbufsize` operand described in 5.4 *Transmission environment definition*.

- When the `key` clause is specified in the extraction definition

$244 + 16 \times \text{number of columns} + \text{data length}^{\#} \text{ (bytes)}$
--

- When the `ukey` clause is specified in the extraction definition

$244 + 16 \times \text{number of columns} + \text{data length}^{\#}$ $+ (32 + 16 \times \text{number of mapping key columns} + \text{sum of the data lengths of mapping keys})$ <p style="margin: 0;">(bytes)</p>

#

Sum of the lengths of the data for all columns subject to linkage (bytes)

4.7 Designing a target Datareplicator

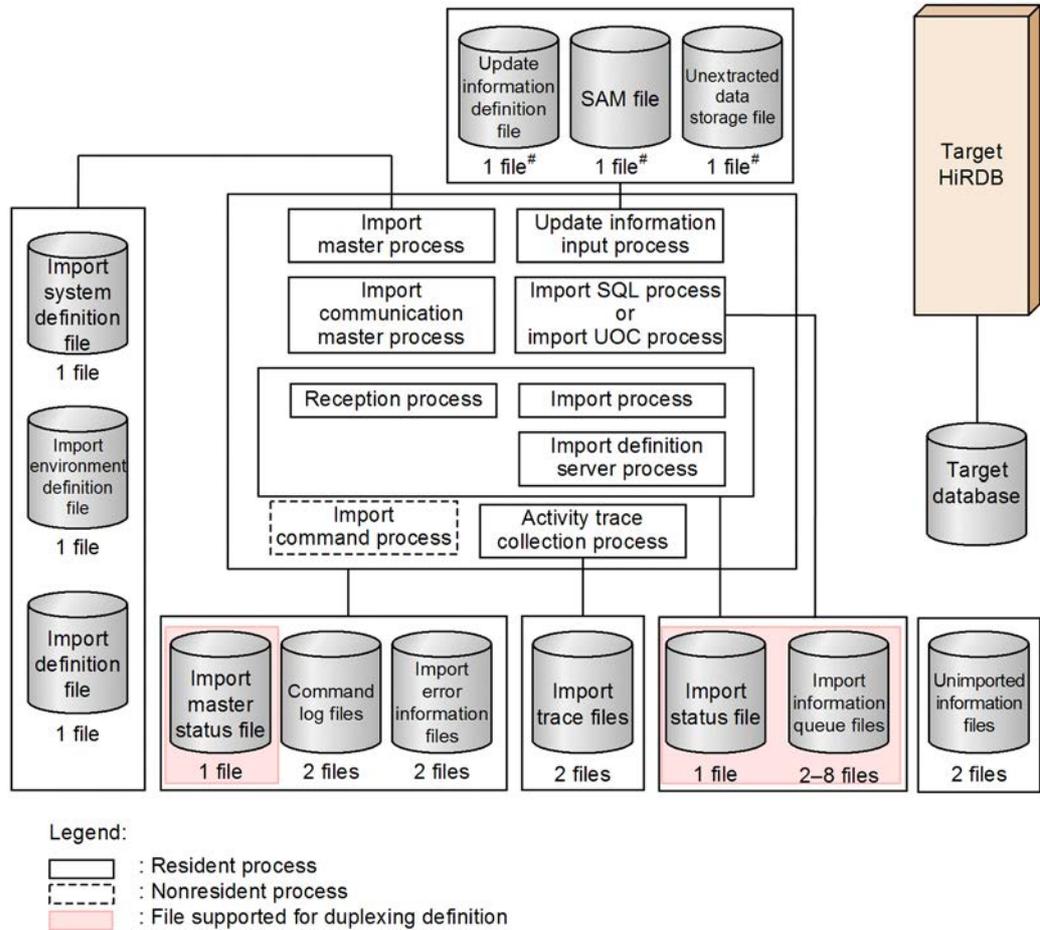
This section explains the system design for a target Datareplicator.

4.7.1 Target Datareplicator's file organization

The figure below shows the target Datareplicator's file organization.

For details about the target Datareplicator's directory structure in UNIX, see 2.3.3 *Directory structure of a target Datareplicator*; for details about the target Datareplicator's directory structure in Windows, see 2.7.3 *Directory structure of a target Datareplicator*.

Figure 4-39: Target Datareplicator's file organization



#

This file is required when data is received from a source system that uses SAM files. However, the update information definition file is not required in RDB1 E2.

4.7.2 Preparation of the files used with the target Datareplicator

This section explains the preparation of the following files, which are needed to use the target Datareplicator:

Files the user must create	Files created by the target Datareplicator during initial startup
<ul style="list-style-type: none"> • Import system definition file • Import environment definition files • Import definition files • Duplexing definition file 	<ul style="list-style-type: none"> • Import information queue files • Import status files • Import master status file • Import error information files • Activity trace files • Import trace files • Unimported information files • Command log files • Duplexing control file

When the source database uses SAM files (PDM2 E2 or RDB1 E2), the user must also prepare the following files, in addition to the files listed above:

- Update information definition file
- SAM file
- Unextracted data storage file

For details about the files, see *3.3.2 Files and processes used during import processing*.

For details about the handling of the files, see *6.7.2 Handling of the files used with the target Datareplicator*.

The following table provides information about creating the files used with the target Datareplicator.

Table 4-40: Creating the files used with the target Datareplicator

File name		File type ^{#8}		Number of files	Required/ optional
		R ^{#9}	C ^{#10}		
Definition files	Import system definition file ^{#1}	Y	N	1 per target system	Required
	Import environment definition file ^{#1}	Y	N	1 per data linkage identifier	Required
	Import definition ^{#1}	Y	N	1 per data linkage identifier	Optional ^{#11}
	Update information definition file ^{#2}	Y	N	1 per update information input command execution	Optional ^{#12}
	Duplexing definition file ^{#1}	Y	N	1 per target system (1 per MGR if the source HiRDB is a parallel server)	Optional
Import information queue file ^{#3, #13}		Y	Y	2 to 8 per data linkage identifier	Required
Status files	Import status file ^{#3, #13}	Y	Y	1 per data linkage identifier	Required
	Import master status file ^{#3, #13}	Y	Y	1 per target system	Required
Unimported information file ^{#4, #5}		Y	N	2 per data linkage identifier	Required
Import error information file ^{#4, #5}		Y	N	2 per target system	Required
Activity trace file (Import trace file)		Y	N	2 per target system	Optional
Command log file ^{#5}		Y	N	2 per target system	Optional
SAM file ^{#6}		Y	N	1 per update information input command execution	Optional ^{#12}
Unextracted data storage file ^{#7}		Y	N	1 per data linkage identifier	Optional ^{#12}
Duplexing control file ^{#4}		Y	N	1 per target system	Optional

Legend:

MGR: System manager

R: UNIX regular file or Windows file

C: UNIX character special file

Y: Can be created.

N: Cannot be created.

#1

Use an OS editor to create this file before starting the target Datareplicator.

#2

Use an OS editor to create this file before starting update information input processing (by executing the `hdssamqin` command) and import processing.

#3

Before you start the target Datareplicator, execute the `hdsstart -i` command to initialize the target Datareplicator. These files are created when the `hdsstart -i` command is executed. If the files are character special files for UNIX, create a symbolic link to the character special files before you start the target Datareplicator.

#4

These files are created automatically when the source Datareplicator is initialized.

#5

These files are created automatically when the source Datareplicator is started.

#6

The SAM file created by the system containing the mainframe database that uses SAM files is transferred to the target Datareplicator by using the mainframe's file transfer program.

#7

When the `hdssamqin` command is executed, the file is created or re-created. If the file already exists, it is re-created with a size of zero bytes.

#8

Use the same file type for all the following files:

- Import information queue files
- Import status file

#9

If the user creates the file (definition file), grant the `read` privilege to source Datareplicator users. Grant the `write` privilege as appropriate. For a file that is not created by the user (definition file), do not change the privilege.

#10

An OS command is used to create character special files. If you use a character

special file, grant the `read` and `write` privileges to target Datareplicator users. If you use the system switchover facility, create the files in the character special file format.

If you have created the import information queue file and import status file as character special files, execute initial start on the target Datareplicator by using the `hdsstart -i -f` command.

When you configure the environment, we recommend that you execute the `hdsstart -i` command with `init` specified to check file capacity so that a shortage of file capacity will not occur while the target Datareplicator is running.

#11

This file can be omitted if the source and target tables have an identical format including the table name and column names.

#12

When SAM files are used for data linkage, this file is used to execute the update information input command (`hdssamqin`). In the case of RDB1 E2, the update information definition file is not needed.

#13

In UNIX, if the file type is OS regular files, data might not be output in the event of a system failure. Because the source Datareplicator uses the import information queue file, import status file, and import master status file during error recovery, a failure cannot be recovered if no data has been output to these files. We recommend that you use character special files that have high reliability for these files.

The following table describes the settings for the files used by at the target Datareplicator.

Table 4-41: File settings used at the target Datareplicator

File type	File name	Remarks
Import system definition file	<code>\$HDSPATH/hdsenv</code>	<ul style="list-style-type: none"> For the information to be defined, see 5.8 <i>Import system definition</i>.
Import environment definition file	<code>\$HDSPATH/any-name</code>	<ul style="list-style-type: none"> For the information to be defined, see 5.9 <i>Import environment definition</i>. Specify <i>any-name</i> in the <code>refenv001</code> through <code>refenv128</code> operands in the import system definition.
Import definition file	<code>\$HDSPATH/any-name</code>	<ul style="list-style-type: none"> For the information to be defined, see 5.10 <i>Import definition</i>. Specify <i>any-name</i> in the <code>reffile</code> operand in the import environment definition.

File type	File name	Remarks
Update information definition file	<i>any-directory/ any-name</i>	<ul style="list-style-type: none"> For the information to be defined, see 5.11 <i>Update information definition</i>. Specify <i>any-name</i> when you execute the <code>hdssamqin -n</code> command.
Duplexing definition file	<i>any-directory/ any-name</i>	<ul style="list-style-type: none"> For the information to be defined, see 5.12 <i>Duplexing definition (target)</i>. Specify <i>any-name</i> in the <code>file_dupenv</code> operand in the import system definition file.
Import information queue file #	<i>any-directory/ any-name</i>	<ul style="list-style-type: none"> Specify <i>any-name</i> in the <code>qufile001</code> through <code>qufile008</code> operands in the import environment definition. Specify the file size in the <code>queuesize</code> operand in the import environment definition.
Import status file [#]	<i>any-directory/ any-name</i>	<ul style="list-style-type: none"> Specify <i>any-name</i> in the <code>statsfile</code> operand in the import environment definition. Specify the file size in the <code>statssize</code> operand in the import environment definition.
Import master status file #	<code>\$HDSPATH/ hdsinitstate</code>	--
Unimported information file	<i>any-directory/ any-name</i>	<ul style="list-style-type: none"> Specify <i>any-name</i> in the <code>unreffile1</code> and <code>unreffile2</code> operand in the import environment definition Specify the file size in the <code>unreffilesz</code> operand in the import environment definition.
Import error information file	<code>\$HDSPATH/ errfile1 \$HDSPATH/ errfile2</code>	<ul style="list-style-type: none"> Specify the file size in the <code>errfilesz</code> operand in the import system definition.
Activity trace file (Import trace file)	<code>\$HDSPATH/ reftrc.trc1 \$HDSPATH/ reftrc.trc2</code>	<ul style="list-style-type: none"> Specify the file size in the <code>int_trc_trcfilesz</code> operand in the import system definition. The file can be edited and referenced by using the <code>hdstrcredit</code> command.
Command log file	<i>any-directory/ any-name</i>	--

File type	File name	Remarks
SAM file	<i>any-directory/ any-name</i>	<ul style="list-style-type: none"> Specify <i>any-name</i> when you execute the <code>hdssamqin -n</code> command. Specify the file size in the <code>int_trc_trcfilesz</code> operand in the extraction system definition. If <code>true</code> is specified in the <code>errfile_unique</code> operand in the extraction system definition, <code>exttrc.trc1_host-name</code> and <code>exttrc.trc2_host-name</code> become the file names. The file can be edited and referenced by using the <code>hdstrcredit</code> command.
Unextracted data storage file	<i>any-directory/ unextfile_data-link age-identifier</i>	--
Duplexing control file	<code>\$HDSPATH/ hds_fileenv.prp</code>	--

Legend:

--: Not applicable

#

This file can be duplexed. However, if a file system area is used, the file cannot be duplexed.

4.7.3 Designing the import procedure

You must design the following items as the data import procedure:

- Import processing method
- Import processing method when the multi-FES facility is used
- DISCONNECT-issuance interval for import processing
- Use of the event facility for automatic control of import processing
- COMMIT-issuance interval for import processing
- Handling of update information that is not defined in the import definition
- Checking for tables subject to import processing
- Size of shared memory for storing definition information

(1) Designing the import processing method

The provided import methods are the *transaction-based import method* and the

table-based import method. For details about these import methods, see 3.3.3 *Import methods.*

You use the `startmode` or `breakmode` operand in the import environment definition to specify the import processing method.

(a) Transaction-based import method

The transaction-based import method imports update information into the target HiRDB database in the order the transactions were updated in the source database. If import of update information is into a table with the same format (same table name, column names, and attributes), you can omit the import definition when you use the transaction-based import method.

(b) Table-based import method

This method creates an import group for one or more tables subject to import processing and imports data for one group at a time. You can define a maximum of 128 import groups per round of import processing. You use the import group definition to specify import groups. For details about the import group definition, see 5.10.6 *Import group definition.*

The table-based import method is broken down into the following types:

- Table-based partitioning method
- Key range-based partitioning method
- Hash partitioning method

Table-based partitioning method

The table-based partitioning method classifies the transactions updated in the source database into user-defined target groups, and then imports them in parallel.

If you use the table-based partitioning method for import processing and there is a referential constraint among tables, group together those tables that have the referential constraint.

Key range-based partitioning method

The key range-based partitioning method imports the transactions updated in the source database in parallel on the basis of user-defined key ranges. Note the following when you use the key range-based partitioning method to perform import processing:

- You can define a maximum of eight key ranges per import group.
- In the key range partitioning condition statement, you can specify only column names in the table subject to import processing that correspond to the mapping key of the table subject to extraction.
- You can specify a maximum of eight conditional statements for one key range partitioning condition. When you specify multiple conditional statements for one

key range partitioning condition, the target Datareplicator connects all the specified conditional statements with AND, enabling specification of complex key range partitioning conditions. If you specify complex key range partitioning conditions, performance might be degraded because of the amount of time required to check the conditional statements. If performance is more important than partitioning, specify fewer conditional statements for a single key range partitioning condition.

- If import processing is concentrated in one key range, performance might not be as good as when key range partitioning is not used. This is because the processing is not distributed in the same manner as when key range partitioning is not used and because the key range partitioning must be checked. In this case, terminate the target Datareplicator normally or immediately, further subdivide the key range in which import processing is concentrated, and then restart the target Datareplicator.
- If you start many import processes or SQL processes while the CPU performance is low, the target Datareplicator might terminate abnormally due to the machine's workload. To avoid this, reduce the number of processes started at one time by using a condition such as `other` to combine key ranges where there is little import processing.

Hash partitioning method

The hash partitioning method uses the hash method to import the transactions updated in the source database in parallel.

(2) Designing the import processing method when the multi-FES facility is used

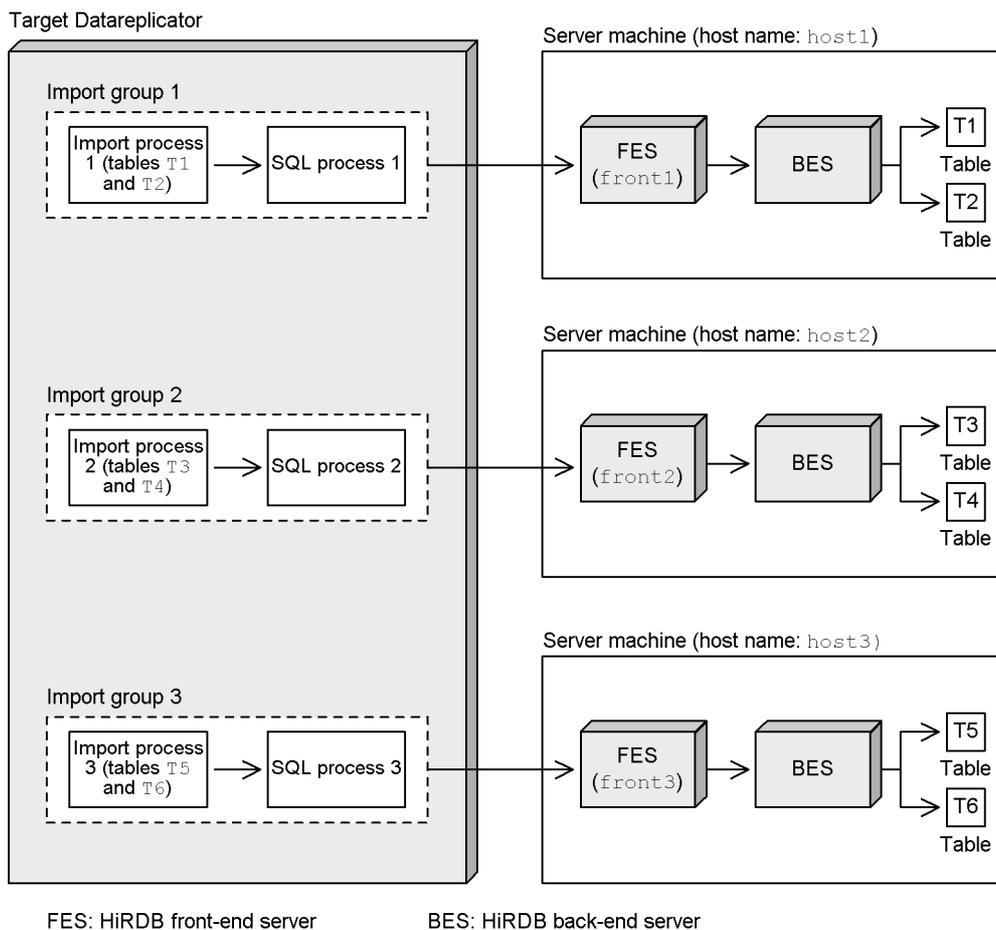
If the target HiRDB uses the multi-FES facility, the target Datareplicator can execute import processing supporting the multi-FES facility. To use the multi-FES facility, you must specify in the import definition the target front-end servers that correspond to the import groups. The target Datareplicator establishes the correspondence to an SQL process for each target front-end server according to the import definition. This enables you to issue SQL statements in parallel for the various target front-end servers, thereby distributing the workload among the front-end servers.

The following provides an example of using the multi-FES facility for each partitioning type under the table-based import method and discusses considerations concerning use of the multi-FES facility.

(a) Table-based partitioning method is used

If tables are grouped by server at the target HiRDB, using the table-based partitioning method enables you to issue the SQL statements in parallel for the various target front-end servers. You can expect more of an improvement in throughput than with the other methods. The following figure provides an example of using the multi-FES facility with the table-based partitioning method.

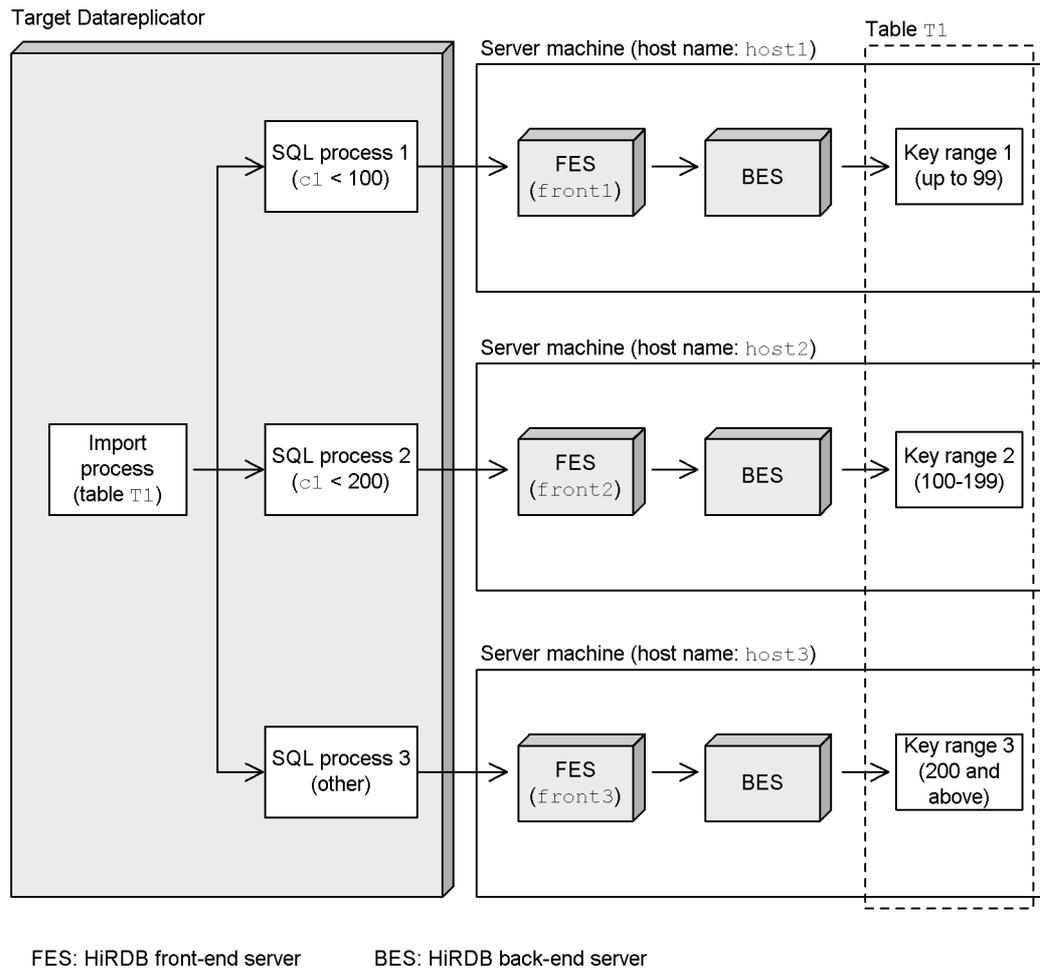
Figure 4-40: Example of using the multi-FES facility with the table-based partitioning method



(b) Key range-based partitioning method

If a table is stored in different servers by row-partitioning key ranges at the target HiRDB, using the key range-based partitioning method enables you to issue the SQL statements in parallel for the various target front-end servers. You can expect an improvement in throughput. The following figure provides an example of using the multi-FES facility with the key range-based partitioning method.

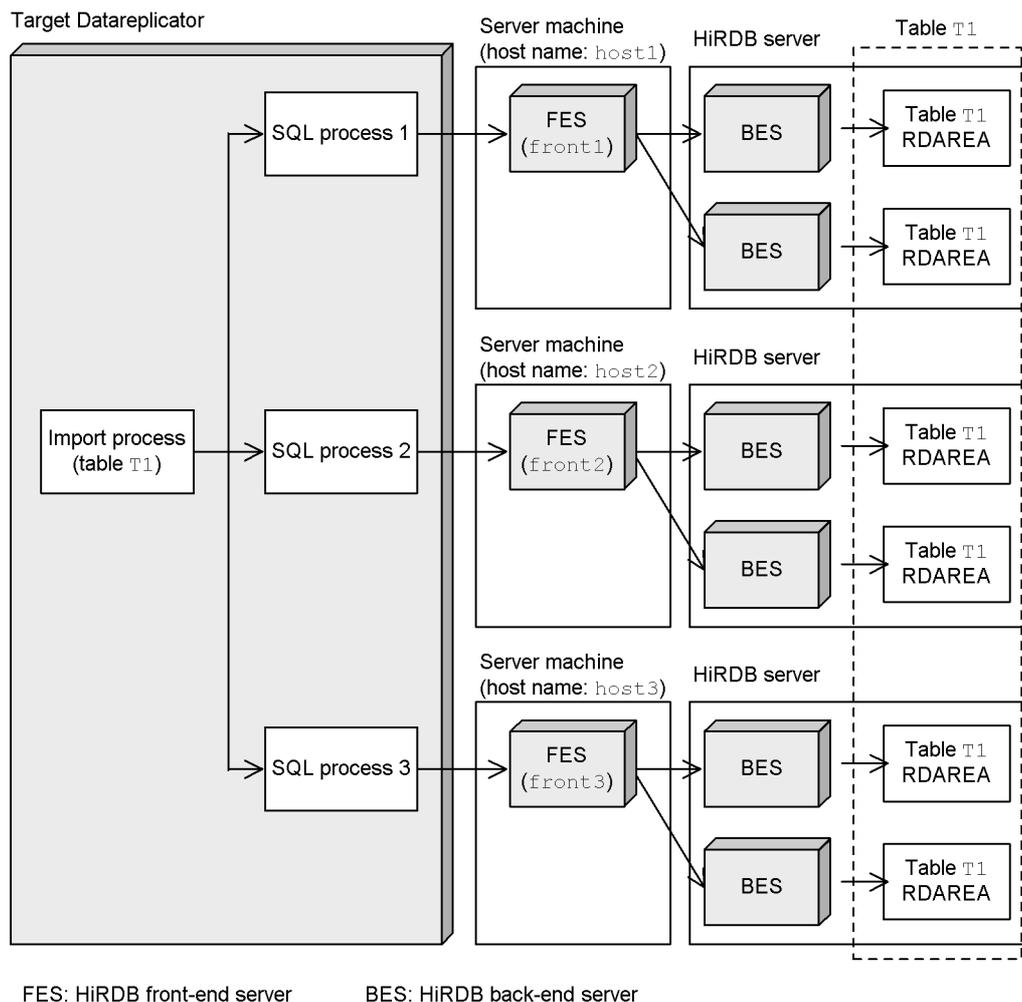
Figure 4-41: Example of using the multi-FES facility with the key range-based partitioning method



(c) Hash partitioning method

If a table is partitioned at the target HiRDB, using the hash partitioning method enables you to execute the SQL statements in parallel for the various front-end servers in accordance with the hash method. Compared to the key range-based partitioning method, this method can execute the processing faster for each front-end server for the multi-FES facility. The following figure provides an example of using the multi-FES facility with the hash partitioning method.

Figure 4-42: Example of using the multi-FES facility with the hash partitioning method



If you employ a multi-FES configuration, the hash partitioning method enables you to set the front-end server to be used to process import data for each partition. This helps to distribute the workload among the front-end servers, and if the front-end server and the back-end server containing the RDAREA are on the same server, it can also reduce the overhead of communication between the front-end server and back-end server.

(d) Considerations

In the case of a multi-FES environment at the target HiRDB where a target front-end

server specified in the import definition and the back-end server that actually executes the import processing are located on different machines, effective processing cannot be executed because of the communication that is required between the front-end server and the back-end server each time import processing occurs. Therefore, if you specify a front-end server in the import definition, specify one that is located on the machine containing the table data.

(3) Designing the DISCONNECT-issuance interval for import processing

You must design the interval at which Datareplicator is to issue DISCONNECT requests to the target HiRDB after it detects the end of the update information in the import information queue file. You use the `disconnect` operand in the import system definition to specify the DISCONNECT-issuance interval for import processing.

Consider the following points in specifying the DISCONNECT-issuance interval:

- If DISCONNECT is not to be issued, specify 0 for the interval.
- If the source database is a HiRDB, when you specify the `discintvl` operand, take into account the value of the `sendintvl` operand in the source Datareplicator's transmission environment definition and the frequency of transaction occurrences in the source system. If the source database is a mainframe database, when you specify the `discintvl` operand, take into account the value of the `RINTERVAL` clause in the XDM/DS startup definition and the frequency of transaction occurrences in the source system.
- If transactions occur frequently at short intervals until the source system's application has terminated, specifying a value close to 0 will increase the probability of DISCONNECT being issued. If the frequency of transaction occurrences fluctuates, you can optimize DISCONNECT issuances by reducing the value of the `sendintvl` operand or the `RINTERVAL` clause if the current value is large, or by increasing the value of the `sendintvl` operand or the `RINTERVAL` clause if the current value is small.

(4) Designing the use of the event facility for automatic control of import processing

You can use the event facility to implement import operations on the basis of events at the source system. To use the event facility, you define event codes to correspond to actual events that are issued at the source system, and you specify these event codes in the source Datareplicator's import environment definition. The following discusses the import operations that can be implemented by the event facility.

(a) Import operations that can be implemented by the event facility

The following table shows the import operations that can be implemented by the event facility and their relationship to the import environment definition.

Table 4-42: Import operations that can be implemented and their relationship to the import environment definition

Type of event	Import operation	Operand to be specified
Import processing stop event	Stops import processing.	eventspd
Transaction-based import event	Switches the import method to the transaction-based import method during import processing.	eventtrn
Table-based import event	Switches the import method to the table-based import method during import processing.	eventtbl
Transaction-based import restart event	Restarts import processing using the transaction-based import method while import processing is stopped.	eventretrn
Table-based import restart event	Restarts import processing using the table-based import method while import processing is stopped.	eventretbl
Event to reset the import processing count	Resets the target Datareplicator's import processing count.	eventcntreset

(b) Considerations in using the event facility

- Use the applicable operand in the import environment definition to specify an event code to correspond to the actual event code that is issued by the source system
- If the source system issues an event that does not correspond to any of the event codes specified in the import environment definition, Datareplicator issues a message to the target system's syslog file. You can use such a message to report the source system's action to the target system. For example, when the source system is terminated, you can report this to the target system by having the source system issue an event that is not specified at the target system.

(5) Designing the COMMIT-issuance interval for import processing

The interval for issuing COMMITs is specified in terms of a number of transactions at the source system. You use the `cmtintvl`, `trncmtintvl`, or `tblcmtintvl` operand in the import environment definition to specify this interval for issuing COMMITs to the target HiRDB.

Consider the following points in specifying the COMMIT-issuance interval:

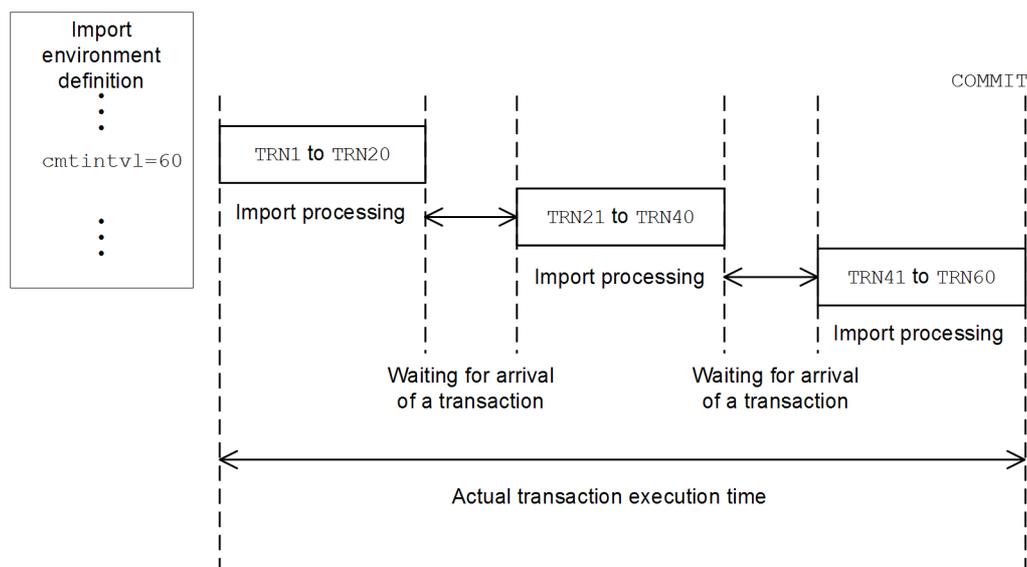
- As the COMMIT-issuance interval increases, SQL statements can be issued to more transactions at one time, thereby reducing the target HiRDB's workload and improving performance. However, if an error occurs, there will be more transactions that cannot be imported and which will be imported automatically the next time import processing starts; thus, if the COMMIT-issuance interval is large,

a considerable amount of time is required for recovery.

- Evaluate the amount of log information at the target HiRDB that will be output during one commit interval during import processing and check that the files needed for error recovery (such as the system log file) will not become filled up.
- A COMMIT is issued whenever the target HiRDB issues the PURGE TABLE SQL statement. Whenever an event is detected, the target Datareplicator issues a COMMIT to obtain a synchronization point. Therefore, the COMMIT interval specified in the import environment definition might not always take effect when update information, including PURGE TABLE, is imported or when events are detected.
- You can specify a COMMIT issuance interval for each import method. Use the `trncmtintvl` operand for the transaction-based import method and the `tblcmtintvl` operand for the table-based import method. To specify a COMMIT-issuance interval common to both import methods, use the `cmtintvl` operand.
- If large values are specified in the `cmtintvl`, `trncmtintvl`, and `tblcmtintvl` operands, the HiRDB server is affected as described below. Make sure that the size of a transaction that is generated from the target Datareplicator will not exceed the range permitted for the target HiRDB.
 - The amount of locked resources (locked rows) increases, resulting in a shortage of locked database resources.
 - The maximum skip count for effective synchronization point dump processing is exceeded and the import processing is rolled back.
 - A commit is issued when the `commit_wait_time` operand value is exceeded, resulting in termination of import processing.

Note:

The execution time of a transaction that is generated from the target Datareplicator is the amount of time required for executing the actual SQL statements plus the amount of time waiting for the arrival of the next transaction to be imported. The following figure shows the actual transaction execution time.



Use the `commit_wait_time` operand to specify the maximum amount of time to wait for the arrival of the next transaction to be imported. If no transaction arrives before the `commit_wait_time` operand value is exceeded, the target Datareplicator automatically issues `COMMIT` and settles the transaction.

(6) Designing the handling of update information that is not defined in the import definition

In the case of update information that is not defined in the import definition, you can design Datareplicator to assume the corresponding source system's extraction definition as the import definition and execute import processing. You use the `defmerge` operand in the import environment definition to specify the handling of update information that is not defined in the import definition.

Note the following point about specifying the handling of update information that is not defined in the import definition:

- Whenever an import definition is not found, Datareplicator assumes unconditionally the extraction definition as the import definition.

(7) Designing checking for tables subject to import processing

You can design the source Datareplicator so that whenever it starts it checks the target HiRDB for tables subject to import processing. You use the `tblcheck` operand in the import environment definition to specify checking for tables subject to import processing.

- If the import definition is omitted, Datareplicator checks the target system for tables subject to extraction processing that are defined in the source system's extraction definition.
- If an import definition is provided, Datareplicator checks the target system for tables subject to the import processing specified in the import definition, regardless of whether the checking discussed here is specified.

(8) Designing the size of shared memory for storing definition information

When the target Datareplicator starts, it allocates a space in shared memory for storing definition information. You use the `defshmsize` operand in the import environment definition to specify the size of the shared memory to be used for storing definition information.

Consider the following point in determining the size of the shared memory for storing the definition information:

- Datareplicator uses the shared memory to store definition information in order to retain the source system's extraction definition information that is stored in the import information queue file. Therefore, you must ensure that the size of the shared memory for storing definition information is greater than the size of the source system's extraction definition information. Otherwise, an error will occur when the target Datareplicator starts. For details about the extraction definition information that is stored in the import information queue file, see 4.7.7

Designing the target Datareplicator's resources.

(9) Designing the resources of the target HiRDB

(a) Number of locked resources

Of the two locked resources described below, use whichever is larger as the estimate for the resources for the target HiRDB. If data linkage is applied to `pdload` that is executed on the source HiRDB, all instances of this statement are replaced with `INSERT` statements at the target. This means that locked resources are required for each `INSERT` statement. Therefore, if a large amount of HiRDB's locked resources are used, the import processing on the target Datareplicator might result in an SQL error due to a shortage of locked resources. If you apply data linkage to `pdload`, estimate the number of locked resources of the target database to be equal to or greater than the number of insert operations performed by `pdload`.

- Enough locked resources for executing the `DESCRIBE` statement on all target tables defined in the `load` statement
- Enough locked resources for executing the SQL statements (`INSERT`, `UPDATE`, and `DELETE`) that are issued between the commits of import processing.

For details about estimating the number of locked resources, see the manual *HiRDB Version 9 System Definition*.

(b) Number of data linkage identifiers that can be connected to HiRDB concurrently

Add the number of data linkage identifier that can be connected to HiRDB concurrently to the `pd_max_users` operand value in the HiRDB definition. The number of data linkage identifiers that can be connected to HiRDB concurrently for each data linkage identifier defined in the import system definition is as follows:

Transaction-based import mode:

One data linkage identifier

Table-based import mode:

Total number of partitions[#] specified in the `group` statements defined in the import definition

#

Use the following for the number of partitions:

- Table-based partitioning method: 1
- Key range-based partitioning method: Number of key range partitions
- Hash partitioning method: Number of hash partitions or SQL partitions

Example:

```
group G1 by T1           : 1
group G2 by T2           : 1
group G3 by T3           : 5
    hash divide into 5
```

In this example, the number of data linkage identifiers that can be connected to HiRDB concurrently is 7 (1 + 1 + 5).

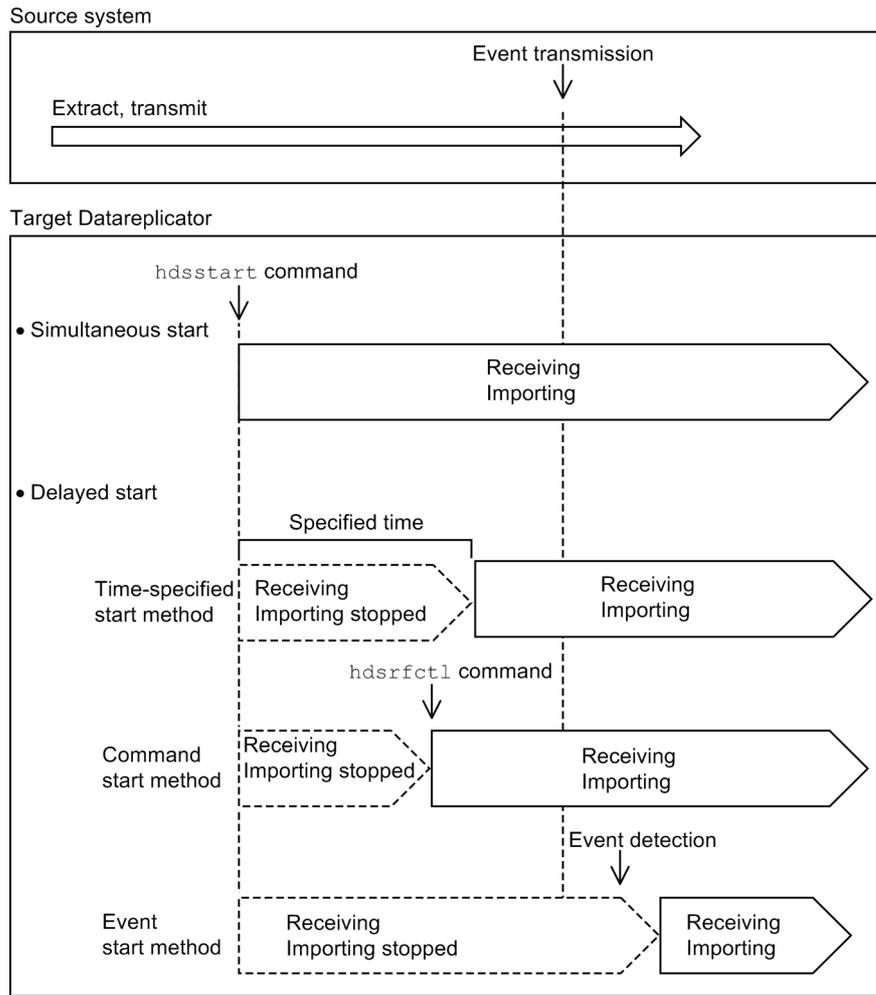
(c) Number of tables that can access HiRDB concurrently

Add the number of tables that can access HiRDB concurrently in the `pd_max_access_tables` operand value in the HiRDB definition. The number of tables that can access HiRDB concurrently is the number of `load` statements specified in the import definition.

4.7.4 Designing the import processing start method

You can design the start method for import processing. The following figure shows the relationship between the import processing start method and the start timing.

Figure 4-43: Relationship between the import processing start method and the start timing



Legend:

-  : Receiving and importing are both underway.
-  : Only importing is stopped.

(1) Simultaneous start

This method starts import processing when the target Datareplicator starts. It enables you to import update information in the order it is received.

- Purpose

This method is appropriate for importing update information in the order it is received.

- Specification in the import environment definition

Specify the `trn` (transaction-based import method) or `tbl` (table-based import method) in the `startmode` operand.

- Command execution

Execute the `hdsstart` command.

(2) *Delayed start*

This method starts only reception processing when the target Datareplicator starts and starts import processing at a specified time (delayed start). Use this method to perform some tasks before starting the import processing, such as changing definitions or maintaining the import database.

When the delayed start method is used, the update information sent from the source system is stored in the import information queue file because reception processing has already begun. There are three types of delayed start:

(a) **Time-specified start method**

This method starts import processing when a user-specified interval elapses. The specified value is an amount of time since the target Datareplicator starts.

- Purpose

This method is appropriate when you must conduct periodic database maintenance after starting the target Datareplicator.

- Specification in the import environment definition

- Specify `spd` in the `startmode` operand.

- Use the `breaktime` operand to specify the amount of time after which import processing is to be started.

- Command execution

Execute the `hdsstart` command.

(b) **Command start method**

This method starts import processing when a command executes. You can use this method to control import operations from the target system.

- Purpose

This method is appropriate when the import start time varies from day to day.

- Specification in the import environment definition
Specify `spd` in the `startmode` operand.
- Command execution
Execute the `hdsrfctl` command.

(c) Event start method

This method starts import processing when an event sent from the source system is detected. You can use this method to control import operations from the source system.

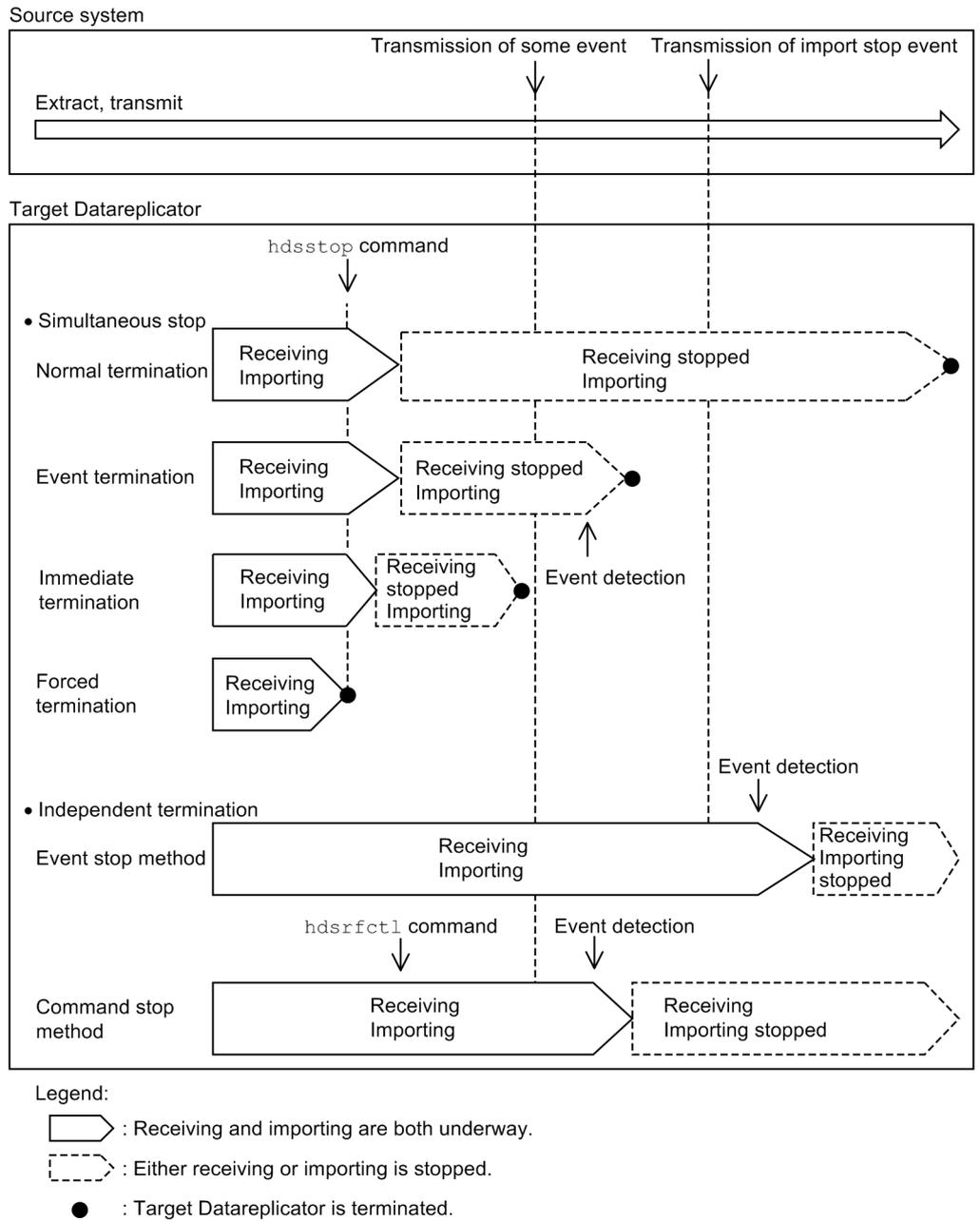
- Purpose
This method is appropriate when you want to use source system operations to control import operations.
- Specification at the source system
Immediately before the start of import processing, specify as an event the UAP that is subject to extraction.
- Specification in the import environment definition
Specify `spd` in the `startmode` operand. Use the `eventretrn` or `eventretbl` operand to establish correspondence between an event code defined at the source system and the import start method to be used after the event is received.

4.7.5 Designing the import processing stop method

You can design the stop method for import processing. The two import processing stop methods are *simultaneous stop* and *independent stop*.

The following figure shows the relationship between the import processing stop method and the stop timing.

Figure 4-44: Relationship between the import processing stop method and the stop timing



(1) Simultaneous stop

This method stops both reception and import processing and terminates the target Datareplicator. When the simultaneous stop method is used, the update information that is sent from the source system is not received because reception processing stops. The target Datareplicator's termination timing depends on the termination mode, as described below.

(a) When the termination mode is normal termination

Import processing stops and the target Datareplicator terminates when import processing has been completed through the end of the source system's operation unit that is active when the `hdsstop` command is executed.

If the source database is a HiRDB, the operation unit is from normal startup to normal termination of the entire target source system. For details about the operation unit when the source database is a HiRDB, see the *VOS3 XDM/DS* manual.

- Purpose

This method is appropriate when you want to terminate the source system or target system normally.

- Specification and operation at the source system

Terminate the source system normally.

- Command execution

Execute the `hdsstop` command.

(b) When the termination mode is event termination

Import processing stops and the target Datareplicator terminates when a specific event sent from the source system is detected after executing the `hdsstop -t event` command. However, if import processing has been completed through the end of the source system's operation unit before the event is detected, import processing will have already terminated before the event is detected.

If the table-based import method is being used, import processing stops when the last group of import processes is completed among all active import processes. When the source Datareplicator has been terminated by the event termination mode, import processing starts from the previous point of termination the next time the target Datareplicator is started.

- Purpose

This method is appropriate when you want to terminate the target system intentionally.

- Specification and operation at the source system

If the source database is a HiRDB, execute the `hdeevent` command at the source

Datareplicator. If it is a mainframe database, specify as an event the UAP that is subject to extraction; make this specification immediately before the intended termination of import processing.

- Command execution

Execute the `hdsstop -t event` command.

(c) When the termination mode is immediate termination

Import processing stops and the target Datareplicator terminates when import of the update information currently being processed is completed after execution of the `hdsstop -t immediate` command.

If the table-based import method is being used, groups of active import processes stop sequentially beginning with the one furthest along in import processing, and import processing stops when the last group reaches the first group's termination point.

After the command has executed, if new update information is received before the last group of import processing terminates, reception processing stops when the update information is stored in the import information queue file.

- Purpose

This method is appropriate when you want to stop the target system temporarily or terminate import processing while reception processing is underway.

- Command execution

Execute the `hdsstop -t immediate` command.

(d) When the termination mode is forced termination

Import processing stops and the target Datareplicator terminates immediately when the `hdsstop -t force` command is executed. If synchronization point processing is underway, import processing stops when the synchronization point processing is completed. When you use the forced termination mode, a loss of conformity might occur between the source and target databases; however, the next time the source Datareplicator starts, the source database will be recovered automatically.

- Purpose

This method is appropriate when you want to terminate the target system immediately.

- Command execution

Execute the `hdsstop -t force` command.

(2) Independent stop

This method stops only import processing, without stopping reception processing. Because reception processing is still underway, update information sent from the

source system is stored in the import information queue file. There are two types of independent stop.

(a) Command termination method

This method stops import processing when the `hdsrfctl` command has executed, and then an event sent from the source system is detected.

- Purpose

This method is appropriate when you want to request termination of import processing from the target system.

- Command execution

Execute the `hdsrfctl` command.

(b) Event stop method

This method stops import processing when an import processing stop event sent from the source system is detected.

- Purpose

This method is appropriate when you want to request termination of import processing from the source system.

- Specification and operation at the source system

If the source database is a HiRDB, execute the `hdeevent` command at the source Datareplicator. If it is a mainframe database, specify as an event the UAP that is subject to extraction; make this specification immediately before the intended termination of import processing.

- Specification in the import environment definition

Use the `eventspd` operand to establish correspondence with an event code specified at the source system.

- Command execution

There is no need to execute a command.

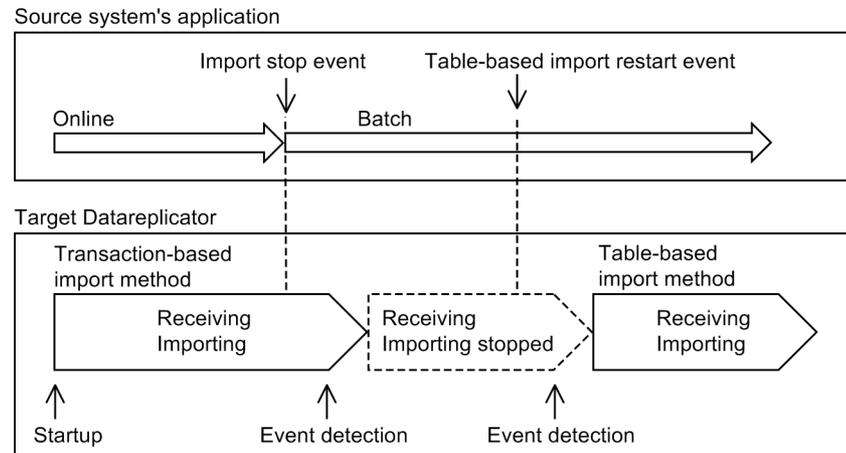
4.7.6 Designing the switching of import processing methods

The target Datareplicator can switch the import processing method on the basis of the source system's application handling. To switch the import processing method, you must establish the correspondence between events sent from the source system and the methods to be used.

If you are using the table-based import method, you can detect events during all import processes by using the message that is issued by the import definition server process when an event is detected. This makes it possible to start the next processing after the

current processes are completed even when multiple import processes are underway. The following figure shows an example of designing to enable switching of the import processing method when the source system switches from an online to a batch application.

Figure 4-45: Example of designing to enable switching of the import processing method



Legend:

-  : Receiving and importing are both underway.
-  : Only importing is stopped

Explanation

- If the source system is executing an online application, the update information is imported sequentially using the transaction-based import method. When the online application terminates, the source system issues an import processing stop event.
- The target system stops import processing when it detects the import processing stop event.
- When batch processing progresses to some degree, the source system issues a table-based import restart event.
- The target system restarts import processing using the table-based import method when it detects the table-based import restart event.

4.7.7 Designing the target Datareplicator's resources

This section explains how to design the target Datareplicator's disk and memory resources.

(1) Designing the target Datareplicator's disk resources

The following table lists the target Datareplicator's disk resources.

Table 4-43: Target Datareplicator's disk resources

File		File type		Required/optional [number of files]	See	Action when file is full
		R	C			
Definition files	Import system definition file	Y	N	Required [1 per import system]	(a)	None
	Import environment definition file	Y	N	Required [1 per data linkage identifier]		
	Import definition file	Y	N	Optional [1 per data linkage identifier]		
	Duplexing definition file	Y	N	Optional [1 per target system]		
Import information queue files ^{#1, #3}		Y	Y ^{#2}	Required [2-8 per data linkage identifier]	(b)	Swapped ^{#1}
Import status file ^{#1, #3}		Y	Y ^{#2}	Required [1 per source system]	(c)	Stops import processing that corresponds to the source system
Import master status file		Y	Y ^{#2}	Required [1 per target system]	(d)	Stops import processing
Unimported information files		Y	N	Required [2 per data linkage identifier]	(e)	Swapped
Import error information files		Y	N	Required [2 per target system]	(f)	Swapped
Activity trace files (import trace files)		Y	N	Optional [2 per target system]	(g)	Swapped
Update information definition files		Y	N	Required [1 per update information input command execution]	(h)	None

File	File type		Required/optional [number of files]	See	Action when file is full
	R	C			
SAM files	Y	N	Required [1 per update information input command execution]	(i)	None
Unextracted data storage file	Y	N	Required [1 per data linkage identifier]	(j)	None
Command log files	Y	N	Optional [2]	(k)	Swapped

Legend:

R: UNIX regular file or Windows file

C: UNIX character special file

Y: Can be created.

N: Cannot be created.

#1

If import processing is not completed on the file to be swapped, the target Datareplicator stops receiving update information until the next transmission interval. If import processing is completed by then, the target Datareplicator starts receiving update information.

#2

If you use character special files with the AIX edition, add 1,024 bytes to the formula for determining the size of each file. In the `queuesize` and `statssize` operands in the import environment definition, specify the size obtained from the formula minus 1,024 bytes.

#3

If you use a Datareplicator file system area, also see *3.5.3(1) Rules for allocating a Datareplicator file system area.*

(a) Sizes of definition files

The sizes of the target Datareplicator's definition files (import system definition file, import environment definition file, and import definition file) depend on the definition operands that are specified.

(b) Size of an import information queue file

Datareplicator stores three types of information in an import information queue file;

these include connection information, extraction definition information, and update information. Each type of information is explained below.

Information stored in the import information queue file

- Connection information

This is the information about the connection between the source system and the target Datareplicator. It is stored only once when connection with the source system is established.

- Extraction definition information

This is the information about the source extraction definition at the source system. It is stored when connection with the source system is established (only when the target Datareplicator is initialized and then started), or when the extraction definition is modified, and then the connection is established with the source system.

- Update information

This is the update information about the source database extracted at the source system. It is stored when update information is received from the source system.

Formulas for determining the size of an import information queue file

The following table shows the formulas for determining the size of an import information queue file.

Table 4-44: Formulas for determining the size of an import information queue file

Item	Formula (bytes)	Description of variables
Connection information	512	None
Extraction definition information	$512 \times \lceil (650 + D + E + F) / 512 \rceil$	<p>$D = 32 + 96 \times \text{number of authorization identifiers} + 64 \times \text{number of extraction definition tables} + 128 \times \text{number of extraction definition columns}$</p> <p>$E = 88 \times \lceil D / \text{update information buffer size}^\# \rceil$</p> <p>$F = 32 \times \lceil (D + E) / 32,453 \rceil$</p>

Item	Formula (bytes)	Description of variables
Update information	$\Sigma (TRN + \Sigma_{INS} + \Sigma_{UPD} + \Sigma_{DEL} + \Sigma_{PRG})$ $TRN = 104$ $INS = Ir Ic$ $Ir = 96 + Row$ $Ic = 80 + (8 \times Cni) + Row$ $UPD = Ur Uc$ $Ur = 96 + Row$ $Uc = 90 + (8 \times Cnu) + Cl$ $DEL = 80 + (8 \times Dni) + Dl$ $PRG = 80$	<p>Σ</p> <p>The first Σ indicates the sum of the update information items. The subsequent Σs (with INS, UPD, DEL, and PRG) indicate the sums for each row.</p> <p><i>TRN</i> Transaction information</p> <p><i>INS</i> INSERT rows</p> <p><i>UPD</i> UPDATE rows</p> <p><i>DEL</i> DELETE rows</p> <p><i>PRG</i> PURGE TABLE rows</p> <p> Indicates OR (selection)</p> <p><i>Ir</i> Row interface</p> <p><i>Ic</i> Column interface</p> <p><i>Row</i> Row length</p> <p><i>Cni</i> Number of columns subject to insertion processing</p> <p><i>Ur</i> Row interface</p> <p><i>Ucn</i> Column interface</p> <p><i>Cnu</i> Sum of the number of columns subject to update processing and the number of mapping keys</p>
--	--	<p><i>Cl</i> Sum of the lengths of updated columns and the lengths of mapping key columns</p> <p><i>Dl</i> Length of mapping key</p> <p><i>Dni</i> Number of mapping key columns</p>

#: If the source database is a HiRDB, the size of the update information buffer is the size of the update information (smt_editbufsize operand value in 5.2 *Extraction system definition*, or the editbufsize operand value in 5.4 *Transmission*)

environment definition). For details about the size of the update information buffer when the source database is a mainframe database, see the manual *VOS3 XDM Data Linkage Facility XDM/DS Description and Definition*.

Notes on estimating the size of an import information queue file

Note the following point about estimating the size of an import information queue file:

- Differences in size depending on the import processing handling method

The following table provides notes on size estimation depending on the import processing handling method.

Table 4-45: Notes on the size estimation depending on the import processing handling method

Handling method	Notes
Simultaneous start	<ul style="list-style-type: none"> • Use the amount of data that is accumulated at the peak of transactions as the disk size. • Operation is possible with just two files. However, if import processing only might be stopped for some reason, we recommend that you increase the number of files.
Delayed start or import processing stop	<ul style="list-style-type: none"> • Calculate the size of the disk space by taking into account the amount of data that can be accumulated before import processing is started or restarted and the amount of data that can be accumulated after the start of import processing. • More than two files might be required, depending on the amount of data accumulated, because the maximum size of an import information queue file is 2 GB. However, if you are using both simultaneous start and delayed start or import processing stop, it might be easier to provide somewhat smaller files to avoid a shortage of file space.

(c) Size of the import status file

The following are the formulas for determining the size of the import status file:

The import status file is used to store the extraction definition information that is sent from the source database. Estimate the size of the extraction definition information based on the size of the extraction definition preprocessing file. If the size of the estimated extraction definition information is too small, a file shortage error occurs when update information is sent from the source system.

For UNIX

$$164 \times 1,024 + \text{size of extraction definition information (bytes)}$$

For Windows

$$282 \times 1,024 + \text{size of extraction definition information (bytes)}$$

For a character special file for UNIX whose sector size exceeds 1 kilobyte

$$164 \times \text{SCT_SIZE} + \lceil \text{size of extraction definition information} / \text{SCT_SIZE} \rceil \times \text{SCT_SIZE (bytes)}$$

SCT_SIZE: Sector size (bytes)

(d) Size of the import master status file

This subsection provides the formula for determining the size of the import master status file.

In the case of the import master status file, the required size is allocated when the target Datareplicator is initialized. The file will never become full because the file capacity is increased during the actual replication.

For UNIX or Windows

$2 \times 1,024$ (bytes)

For a character special file for UNIX whose sector size exceeds 1 kilobyte

$2 \times SCT_SIZE$ (bytes)

SCT_SIZE: Sector size (bytes)

(e) Size of an unimported information file

The default size is 16KB. If you want to retain information about SQL errors that might occur, increase the file size.

(f) Size of an import error information file

The default size is 16KB. If you want to retain error information covering a long period of time, increase the file size.

(g) Size of an activity trace file

Use the `int_trc_filesz` operand in the import system definition to specify the size of an activity trace file (import trace file).

(h) Size of the update information definition file

The size of the update information definition file depends on the definition operands that are specified.

(i) Size of a SAM file

The size of a SAM file depends on the size of a file transferred from PDM2 E2 or RDB1 E2.

(j) Size of the unextracted data storage file

The size of the unextracted data storage file depends on the number of data items that are not subject to extraction processing. The size of the unextracted data storage file is increased as much as the space available in the file system.

(k) Size of a command log file

The size is always 128 KB.

(1) Sizes of other files

The Windows edition of Datareplicator creates several work files in the `tmp` directory under the installation directory. Therefore, provide 4 MB of space for the work files.

(2) Designing the target Datareplicator's memory resources

The following table provides a list of memory resources for the target Datareplicator.

Table 4-46: List of memory resources for the target Datareplicator

Memory resource	Subsection
Import master process	(a)
Import communication master process	(b)
Import definition server process	(c)
Import process	(d)
Import SQL process	(e)
Import UOC process	(f)
Update information input process	(g)
Activity trace collection process	(h)
Datareplicator agent process	(i)
Size of shared memory for storing definition information	(j)
Size of shared memory for communication between import processes	(k)
Size of shared memory for importing BLOB columns	(l)
Number of required semaphores	(m)

Table 4-47: List of variables used in the formulas for determining the size of memory for target Datareplicator

Variable name	Description of variable
<i>ATTR_NUM</i>	Total number of abstract data type attributes subject to data linkage
<i>CNST_NUM</i>	Total number of name clauses specified in the <code>const</code> clauses of all <code>format</code> statements specified in the import definition
<i>COL_MAX</i>	Maximum value of the total actual data length of columns subject to extraction per table subject to extraction (BLOB and BINARY columns are excluded) (bytes)

Variable name	Description of variable
<i>COL_MNUM</i>	Maximum number of columns per table in all tables subject to extraction
<i>COL_NUM</i>	Total number of columns in all tables subject to extraction
<i>CUOC_CLM_NUM</i>	Total number of imported columns for which a column data editing UOC routine is specified
<i>DUP_UPD</i>	Total number of duplicate update information names defined per extraction table
<i>ELEM_LEN</i>	Maximum value of the total length of all repetition column elements per table subject to extraction (bytes) (This is the actual length extracted, not the definition length)
<i>ELEM_NUM</i>	Maximum value of the total number of repetition column elements per table subject to extraction (This is the actual number of elements extracted, not the defined number of elements)
<i>EXCS_R</i>	Total number of columns subject to import (including columns subject to import that are assumed when <code>true</code> is specified in the <code>defmerge</code> operand and the <code>load</code> statement is omitted in the import environment definition)
<i>FLD_NUM</i>	Total number of field names specified in all <code>load</code> statements defined in the import definition
<i>FOR_NUM</i>	Total number of <code>format</code> statements specified in the import definition
<i>GRP_NUM</i>	Total number of <code>group</code> statements specified in the import definition
<i>GRPD_NUM</i>	Total number of groups specified in the <code>group</code> statement specified in the import definition
<i>GRPHA_NUM</i>	Total number of hash-partitioned RDAREAs for tables for which the <code>hash</code> clause is specified in the <code>group</code> statement in the import definition
<i>GRPHC_LEN</i>	Total length of hash key column values in tables for which the <code>hash</code> clause is specified in the <code>group</code> statement in the import definition (bytes)
<i>GRPHC_NUM</i>	Total number of hash key columns in tables for which the <code>hash</code> clause is specified in the <code>group</code> statement in the import definition
<i>GRPK_NUM</i>	Total number of key range partitioning conditions in the <code>group</code> statement specified in the import definition
<i>GRPT_NUM</i>	Total number of tables specified in the <code>group</code> statement specified in the import definition

4. System Design

Variable name	Description of variable
<i>IDX_NUM</i>	Maximum number of indexes created per table subject to import
<i>IDXCLM_NUM</i>	Maximum number of index component columns for indexes created per table subject to import
<i>KEY_MAX</i>	Maximum value of the total length of mapping keys per table subject to extraction (bytes)
<i>KEY_MNUM</i>	Maximum value of the total number of mapping keys per table subject to extraction
<i>KEY_NUM</i>	Total number of mapping keys in all tables subject to extraction
<i>LOD_NUM</i>	Total number of <code>load</code> statements specified in the import definition
<i>NAME_NUM</i>	Total number of <code>name</code> clauses in all <code>format</code> statements specified in the import definition
<i>NCHR_MAX</i>	Maximum length of a column definition of national character string attributes (<code>NCHAR</code> , <code>NVARCHAR</code> , and <code>LONG NVARCHAR</code>) subject to extraction (bytes)
<i>PRP_FILE</i>	Size of extraction definition preprocessing file (bytes)
<i>RDEF_FILE</i>	Size of import definition file (bytes)
<i>REP_NUM</i>	Total number of repetition columns in all tables subject to extraction
<i>RMST_FILE</i>	Size of import master status file (bytes)
<i>RQUE_FILE</i>	Size of import information queue file (bytes)
<i>RST_FILE</i>	Size of import status file (bytes)
<i>RUOC_ATTR_NUM</i>	Total number of data attributes in ADT that will be passed to an import information editing UOC routine
<i>RUOC_COL_MAX</i>	Maximum number of import columns that will be passed to an import information editing UOC routine
<i>RUOC_KEY_MAX</i>	Maximum number of mapping key columns that will be passed to an import information editing UOC routine
<i>RUOC_TYPE_NUM</i>	Total number of data types in ADT that will be passed to an import information editing UOC routine
<i>SKIP_LIST</i>	Number of rows in the import suppression list file
<i>SKIPTYPEONLY</i>	Total number of <code>SKIP_TYPE_ONLY</code> clauses specified in the import suppression list file

Variable name	Description of variable
<i>SV_NUM</i>	When the source is HiRDB: Total number of back-end servers subject to extraction When the source is not HiRDB: 1
<i>TBL_NUM</i>	Total number of tables subject to extraction
<i>TBLT_R</i>	Total number of tables subject to import (including tables subject to import that are assumed when <code>true</code> is specified in the <code>defmerge</code> operand and the <code>load</code> statement is omitted in the import environment definition)
<i>TYPE_NUM</i>	Total number of abstract data types in all tables subject to extraction
<i>UBUF</i>	When the source database is HiRDB: <ul style="list-style-type: none"> If <code>nodemst</code> is specified in <code>sendcontrol</code> in the extraction system definition <code>editbufsize</code> value specified in the transmission environment definition If <code>sendmst</code> is specified in <code>sendcontrol</code> in the extraction system definition <code>smt_editbufsize</code> value specified in the extraction system definition When the source database is XDM/DS: <code>REFLECTBUFF</code> value specified in the XDM/DS start definition. For details, see the manual <i>VOS3 XDM Data Linkage Facility XDM/DS Description and Definition</i> (Specify this value in KB)
<i>UPD_NUM</i>	Total number of update information names

(a) Import master process**Number of processes**

There can be only one import master process per target Datareplicator.

Size of procedure

204,800 bytes

Dynamic memory size (bytes)

1,024

(b) Import communication master process**Number of processes**

Number of corresponding data linkage identifiers + 1

Size of procedure

256,000 bytes

Dynamic memory size (bytes)

The dynamic memory size is the sum of the fixed part shown below and the variable part explained below if applicable.

Fixed part:

$$\begin{aligned}
 &UBUF \times 1,024 + 136 \\
 &+ UBUF \times 1,024 \\
 &+ \uparrow(UBUF \times 1,024 + 64) / 1,024 \uparrow \times 1,024 \\
 &+ 216 \\
 &+ 281 \\
 &+ 1,024 \\
 &+ 68,228 \\
 &+ 5,219 \\
 &+ 10,280 + \text{MAX}(1,024, PRP_FILE / SV_NUM) \\
 &+ \text{MAX}(RMST_FILE, RST_FILE, RQUE_FILE) - 1,024 \\
 &+ 512,000
 \end{aligned}$$

Variable part:

- Formula to be added if the duplexing function
65,536

Explanation of the variables

See Table 4-47 *List of variables used in the formulas for determining the size of memory for target Datareplicator.*

(c) Import definition server process

Number of processes

As many import definition server processes are required as there are corresponding data linkage identifiers.

Size of procedure

409,600 bytes

Dynamic memory size (bytes)

The dynamic memory size is the sum of the fixed part shown below and any applicable variable part explained below.

Fixed part:

$$\begin{aligned}
 & (\uparrow(UPD_NUM + LOD_NUM) / 16 \uparrow \times 16) \times 160 \\
 & + (\uparrow(COL_NUM + FLD_NUM) / 16 \uparrow \times 16 + TBL_NUM) \times 128 \\
 & + \uparrow(PRP_FILE / SV_NUM + 152) / 1,024 \uparrow \times 1,024 \\
 & + PRP_FILE / SV_NUM \\
 & + \uparrow(PRP_FILE / SV_NUM + 64) / 1,024 \uparrow \times 1,024 \\
 & + PRP_FILE / SV_NUM \\
 & + \uparrow(PRP_FILE / SV_NUM + 64) / 1,024 \uparrow \times 1,024 \\
 & + (\uparrow COL_NUM / (TBL_NUM \times 16) \uparrow + TBL_NUM) \times 384 \\
 & + \uparrow TBLT_R / 16 \uparrow \times 704 \\
 & + 260 \\
 & + 68,228 \\
 & + 5,219 \\
 & + 10,280 + \text{MAX}(1,024, PRP_FILE / SV_NUM) \\
 & + \text{MAX}(RMST_FILE, RST_FILE, RQUE_FILE) - 1,024 \\
 & + \uparrow UPD_NUM / 16 \uparrow \times 704 \\
 & + 512,000
 \end{aligned}$$

Variable part:

- Formula to be added if the import definition is specified
RDEF_FILE
- Formula to be added if the `format` statement with the `const` clause specified is defined in the import definition
 $\uparrow CNST_NUM / 16 \uparrow \times 832$
- Formula to be added if the `format` statement is specified in the import definition
 $(\uparrow \text{Total number of } \text{format} \text{ statements specified in the import definition} / 16 \uparrow \times 16) \times 36$
- Formula to be added if the `format` statement is defined in the import definition
 $\uparrow FOR_NUM / 16 \uparrow \times 576$

$$+ \uparrow NAME_NUM / 16 \uparrow \times 768$$

- Formula to be added if the load statement is specified in the import definition

$$\uparrow LOD_NUM / 16 \uparrow \times 832$$

$$+ \uparrow LOD_NUM / 8 \uparrow \times 704$$

$$+ \uparrow FLD_NUM / 16 \uparrow \times 384$$

- Formula to be added if an asterisk (*) is not specified in the field specification in the load statement

$$\uparrow FLD_NUM / 16 \uparrow \times 896$$

- Formula to be added if the target able specified in the load statement contains character set columns

32

- Formula to be added if the group statement is specified in the import definition

$$\uparrow (GRP_NUM + 2) / 16 \uparrow \times 2,752$$

$$+ (\uparrow GRPT_NUM / 16 \uparrow + GRP_NUM) \times 64$$

- Formula to be added if groups are specified in the group statement

$$\uparrow (GRPD_NUM + 2) / 16 \uparrow \times 1,024$$

- Formula to be added if the following conditions are true:

not_null_unique or unique is specified in the mapping_key_check clause in the import environment definition.

not_null_unique or unique is specified in the check clause of the LOAD statement in the import definition

$$8 \times IDX_NUM + 32 \times IDXCLM_NUM$$

$$+ 16 + 4 \times IDX_NUM$$

$$+ 580$$

- Formula to be added if key range partitioning conditions are specified in the group statement in the import definition

$$(\uparrow GRPK_NUM / (GRPD_NUM \times 16) \uparrow + GRPD_NUM) \times 4,416$$

$$+ 32$$

- Formula to be added if columns with character set specification are linked

32

- Formula to be added if abstract data-type columns are linked

$$64 + 112 \times TYPE_NUM + 64 \times ATTR_NUM$$

- Formula to be added if the duplexing function

65,536

Explanation of the variables

See Table 4-47 *List of variables used in the formulas for determining the size of memory for target Datareplicator.*

(d) Import process

Number of processes

When the transaction-based import method is used, only one import process is required. When the table-based import method is used, the number of required import processes equals the maximum number of import groups.

Size of procedure

256,000 bytes

Dynamic memory size (bytes)

The dynamic memory size is the sum of the fixed part shown below and any applicable variable part explained below.

Fixed part:

1,464
 + MAX(48,000, COL_MAX + 4,096)
 + 32
 + 67
 + UBUF x 1,024
 + 60
 + 845 + KEY_MAX
 + 19,632
 + 12 x ↑KEY_MNUM / 10 ↑ x 10
 + 12 x ↑COL_MNUM / 50 ↑ x 50
 + 68,228
 + 5,219

$$\begin{aligned}
 &+ 10,280 + \text{MAX}(1,024, \text{PRP_FILE} / \text{SV_NUM}) \\
 &+ \text{MAX}(\text{RMST_FILE}, \text{RST_FILE}, \text{RQUE_FILE}) - 1,024 \\
 &+ 512,000
 \end{aligned}$$

Variable part:

- Formula to be added if the hash clause is specified in the group statement in the import definition

$$\begin{aligned}
 &\text{GRPHA_NUM} \times 4 \\
 &+ \text{GRPHC_NUM} \times 4 \\
 &+ \text{GRPHC_NUM} \times 4 \\
 &+ \text{GRPHC_NUM} \times 256
 \end{aligned}$$

- Formula to be added if the hash clause is specified in the group statement in the import definition and the hash key columns include at least one column with character set specification

$$\text{GRPHC_NUM} \times 4$$

- Formula to be added if the hash clause is specified in the group statement in the import definition and the hash value is invalid

$$\text{GRPHC_LEN}$$

- Formula to be added if the import suppression function is used

$$\begin{aligned}
 &92 \\
 &+ 52 \times \text{SKIP_LIST} \\
 &+ \uparrow \text{SKIPTYPEONLY} / 10 \uparrow \times 40
 \end{aligned}$$

- Formula to be added if the source database is XDM/DS and data linkage is applied to the NCHAR, NVARCHAR, or LONG NVARCHAR type

$$\text{NCHR_MAX} + 2$$

- Formula to be added if the duplexing function is used

$$65,536$$

Explanation of the variables

See Table 4-47 *List of variables used in the formulas for determining the size of memory for target Datareplicator.*

(e) Import SQL process

Number of processes

When the transaction-based import method is used, only one import SQL process is required. When the table-based import method is used, the number of required import SQL processes equals the maximum number of import groups.

Size of procedure

307,200 bytes

Dynamic memory size (bytes)

The dynamic memory size is the sum of the fixed part shown below and any applicable variable part explained below.

Fixed part:

1,448
 + 88 + 40 x *COL_MNUM* + 39 x *KEY_MNUM*
 + 197 + 40 x *COL_MNUM* + 39 x *KEY_MNUM*
 + 4,096
 + 4,096
 + 68 x *TBLT_R*
 + 32 x *TBLT_R* + 16 x (*EXCS_R* - 1) + 2 x *EXCS_R*
 + 32 x *TBLT_R* + 16 x (*EXCS_R* + *KEY_NUM* - 1) + 2 x (*EXCS_R* + *KEY_NUM*)
 + 32 x *TBLT_R* + 16 x (*EXCS_R* - 1) + 2 x *EXCS_R*
 + 2 x *EXCS_R*
 + 68,228
 + 5,219
 + 10,280 + MAX(1,024, *PRP_FILE* / *SV_NUM*) + MAX(*RMST_FILE*,
RST_FILE, *RQUE_FILE*) - 1,024
 + 512,000

Variable part:

- Formula to be added if the WITH LOCK clause is specified in the LOAD statement in the import definition
 $48 \times \lceil LCKT_NUM / 10 \rceil \times 10$
- Formula to be added if columns with character set specification are linked
 $96 \times TBLT_R + 2 \times EXCS_R$
 $+ 96 \times TBLT_R + 2 \times (EXCS_R + KEY_NUM)$

$$+ 96 \times TBLT_R + 2 \times KEY_NUM$$

- Formula to be added if abstract data-type columns are linked

$$88 + 35 \times COL_MNUM$$

- Formula to be added if repetition columns are linked

$$16 \times (REP_NUM + 1)$$

$$+ (ELEM_LEN + 2 \times ELEM_NUM + 8 \times TBLT_R)$$

- Formula to be added if abstract data-type columns or repetition columns are linked

$$84 + 40 \times COL_MNUM + 39 \times KEY_MNUM$$

- Formula to be added if columns with character set specification are linked

$$52 \times CNST_NUM$$

- Formula to be added if a column data UOC routine is used

$$CUOC_CLM_NUM \times 48$$

$$+ CUOC_CLM_NUM \times 32,008$$

- Formula to be added if one extraction table contains duplicate update information names

$$20 \times \uparrow DUP_UPD / 5 \uparrow$$

- Formula to be added if abstract data-type columns are linked

$$80 \times TYPE_NUM$$

- Formula to be added if the duplexing function is used.

$$65,536$$

Explanation of the variables

See Table 4-47 *List of variables used in the formulas for determining the size of memory for target Datareplicator.*

Note:

If the extraction column types include the BLOB type, calculate the corresponding column as 20 bytes in the calculation of the maximum update row length.

(f) Import UOC process

Number of processes

Only one import UOC process is required when a UOC routine is used.

Size of procedure

(102,400 + *size of UOC routine*) bytes

Dynamic memory size (bytes)

The dynamic memory size is the sum of the fixed part shown below plus any applicable variable part explained below.

Fixed part:

408
 + MAX(48,000, *COL_MAX* + 4,096)
 + 68,228
 + 5,219
 + 10,280 + MAX(1,024, *PRP_FILE* / *SV_NUM*)
 + MAX(*RMST_FILE*, *RST_FILE*, *RQUE_FILE*) - 1,024
 + 512,000
 + 48 x *RUOC_COL_MAX*
 + 48 x *RUOC_KEY_MAX*
 + 824

Variable part:

- Formula to be added if update information names that use an import information editing UOC routine include ADT columns

$120 \times \uparrow \text{RUOC_TYPE_NUM} / 10 \uparrow$
 + 84 x *RUOC_TYPE_NUM* + 60 x *RUOC_ATTR_NUM*

- Formula to be added if the file duplexing function is used

65,536

Explanation of the variables

See Table 4-47 *List of variables used in the formulas for determining the size of memory for target Datareplicator.*

(g) Update information input process**Number of processes**

Only one update information import process is required when data linkage uses a SAM file.

Size of procedure

250KB

Dynamic memory size (KB)

$300 + DBUF + TBUF$

Explanation of the variables

DBUF: 0.5 x total number of fields in the extraction statements in the update information definition

TBUF: Total length of update data per transaction x 2

(h) Activity trace collection process

Number of processes

One process is required for the target Datareplicator.

Size of procedure

60KB

Dynamic memory size

50KB

(i) Datareplicator agent process

Number of processes

Add the number of processes either at the source system or at the target system.

Size of procedure

70KB

Dynamic memory size (KB)

$\uparrow 300,000^{\#} + (HDS_NUM + DC_NUM \times 3 + GRP_NUM) \times 64 / 1,024 \uparrow$

Explanation of the variables

HDS_NUM: Total number of target systems

DC_NUM: Total number of data linkages

GRP_NUM: Total number of import groups

#: Use the value for either the source system or the target system.

(j) Determining the size of shared memory for storing definition information

The following is the formula for determining the size of shared memory for storing definition information that is specified with the `defshmsize` operand in the import environment definition:

$\uparrow 192 \times (GRPT + 2) + 160 \times TBLT_E + 152 \times EXCS_E + 156 \times TBLT_r + 128 \times EXCS_r + 88 \times TBLT_R + 24 \times EXCS_R + 52 \times CNST \uparrow$ (bytes)

GRPT: Number of import group definitions in the import definition file (number of group statements in the import definition)

TBLT_E: Number of extraction tables

EXCS_E: Number of extraction columns

TBLT_r: Number of import table definitions in the import definition file (number of load statements in the import definition)

EXCS_r: Total number of columns in the import table specified in the import table definition

TBLT_R: Number of import tables (including the import tables that are assumed when true is specified in the `defmerge` operand described in 5.9 *Import environment definition* and the `load` statement is omitted)

EXCS_R: Number of import columns (including the import columns that are assumed when true is specified in the `defmerge` operand described in 5.9 *Import environment definition* and the `load` statement is omitted)

CNST: Total number of `const` clauses specified in the update information field definition in the import definition file

(k) Determining the size of shared memory for communication between import processes

$82,292 + 71,556 \times UJ$ (bytes)

UJ: Number of source systems

(l) Determining the size of shared memory for importing BLOB columns

The following shared memory size is required for each import group:

$(LOB_SIZE + 4) \times 4 + 80$ (bytes)

LOB_SIZE: Maximum value in the import group (total length of defined BLOB columns that are included in each update information name)

(m) Determining the number of required semaphores

The following is the formula for determining the number of semaphores used by the target Datareplicator:

$5 + 10 \times UJ$

UJ: Number of source systems

Change the maximum number of semaphores in the entire system that is specified in the kernel parameter (`SEMMNS`) so that the number of semaphores determined by the

4. System Design

above formula can be used at the target system. For details about how to update the kernel parameter, see the applicable OS documentation.

4.8 Designing the source HiRDB

This section describes the system design of the source HiRDB.

When data is to be extracted from a HiRDB database, the source HiRDB adds the data linkage information required by the source Datareplicator to the database update log and outputs it to the system log file. The source Datareplicator executes data linkage on the basis of the database update log information in the system log file that was output by the source HiRDB. The source HiRDB's facility for output of data linkage information to the system log file is called the *HiRDB Datareplicator linkage facility*.

When you start the HiRDB Datareplicator linkage facility at the source HiRDB, the system log containing the data linkage information is output to the system log file. When you start the source Datareplicator to begin extraction processing, the source Datareplicator starts extracting the system log information from the system log file.

To extract data from a HiRDB database, you must do the following for the source HiRDB:

- Specify the name of the directory used by the source Datareplicator
- Specify startup of HiRDB Datareplicator linkage

(1) Specifying the name of the directory used by the source Datareplicator

Specify the name of the directory used by the source Datareplicator in the source HiRDB's unit control information definition. The source HiRDB uses the data linkage file in this directory to report the extraction status to the source Datareplicator. If this directory name is not specified, you cannot use the HiRDB Datareplicator linkage facility. The following shows this specification:

```
pd_rpl_hdepath=source-HiRDB-Datareplicator-directory-name
```

For details about the specification of `pd_rpl_hdepath` in the unit control information definition, see *5.6 Source HiRDB definition*.

(2) Specifying startup of HiRDB Datareplicator linkage

To output data linkage information together with the system log at the source HiRDB, start HiRDB Datareplicator linkage by one of the following methods:

- Specify `pd_rpl_init_start=Y` in the source HiRDB's system common definition

Use this method to start HiRDB Datareplicator linkage when the source HiRDB starts. In this case, the data linkage information that is needed is output to the

system log file from the time the source HiRDB starts.

For details about the specification of `pd_rpl_init_start` in the system common definition, see *5.6 Source HiRDB definition*.

- Execute the `pdrplstart` command (when `pd_rpl_init_start=Y` is not specified in the system common definition or HiRDB Datareplicator linkage has been cancelled by the `pdrplstop` command)

If `pd_rpl_init_start=Y` is not specified in the system common definition, use the `pdrplstart HiRDB` command to start HiRDB Datareplicator linkage.

When the `pdrplstart` is executed, the source HiRDB starts output of data linkage information from the update information for transactions that will be executed after execution of the `pdrplstart` command. If you have already executed a transaction that updates the database that is subject to data linkage before you executed the `pdrplstart` command, conformity between the source and target databases might be lost. Therefore, before you enter the `pdrplstart` command, check that no transactions that update the database subject to data linkage have already executed.

If HiRDB Datareplicator linkage has been cancelled by HiRDB's `pdrplstop` command, execute the `pdrplstart` command to restart data linkage. When HiRDB Datareplicator linkage is cancelled by the `pdrplstop` command, conformity between the source and target databases is lost, and you must re-create the target database before executing the `pdrplstart` command.

For details about how to execute the `pdrplstart` or `pdrplstop` command and how to re-create a target database, see *6.5.6 Source HiRDB handling procedure*.

For details about the formats of the `pdrplstart` and `pdrplstop` commands, see *7. Command Syntax*.

(3) Notes after HiRDB Datareplicator linkage has started

If the source Datareplicator's extraction processing is stopped for an extended period of time after HiRDB Datareplicator linkage started, HiRDB's system log file might become full and HiRDB might shut down. We recommend that you do not stop the source Datareplicator's extraction processing while HiRDB is running.

Chapter

5. Definitions

This chapter explains the procedures for defining the design of a data linkage system between source and target Datareplicators and provides definition examples.

- 5.1 Overview of Datareplicator definitions
- 5.2 Extraction system definition
- 5.3 Extraction environment definition
- 5.4 Transmission environment definition
- 5.5 Extraction definition
- 5.6 Source HiRDB definition
- 5.7 Duplexing definition (source)
- 5.8 Import system definition
- 5.9 Import environment definition
- 5.10 Import definition
- 5.11 Update information definition
- 5.12 Duplexing definition (target)
- 5.13 Examples of Datareplicator definitions

5.1 Overview of Datareplicator definitions

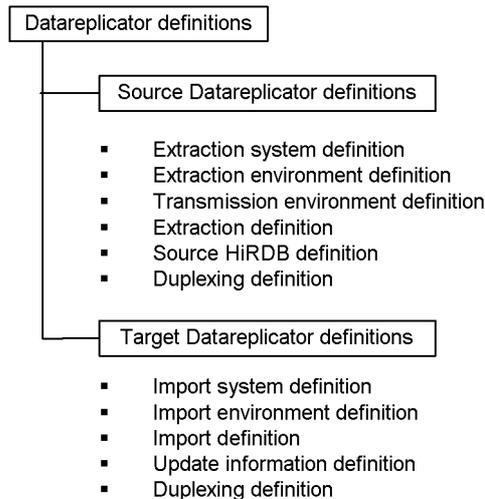
This section provides an overview of the Datareplicator definitions. For details about the information to be defined in a mainframe's data linkage product (XDM/DS), see the *VOS3 XDM/DS* manual.

5.1.1 Organization of the Datareplicator definitions

Datareplicator definition involves creating separate definition files for source and target Datareplicators. You must create both types of definitions before you can start a data linkage system. In addition, you must specify use of Datareplicator for data linkage in the system definition for the source HiRDB.

The following figure shows the organization of the Datareplicator definitions.

Figure 5-1: Organization of the Datareplicator definitions



For details about how to create Datareplicator definition files, see *4.6.2 Preparation of the files used with the source Datareplicator* and *4.7.2 Preparation of the files used with the target Datareplicator*.

Sample definition files

Datareplicator provides templates for the definition files (sample definition files). If you copy the templates into the directory for definition files (HDSPTH) or into some other appropriate directory, and then modify a template's contents, you will not have to create the definition file from scratch. For details about Datareplicator's sample definition files, see *5.13.2 Examples of source Datareplicator definitions* and *5.13.3 Examples of target Datareplicator*

definitions.

Table 5-1 lists the source Datareplicator definitions, and Table 5-2 lists the target Datareplicator definitions.

Table 5-1: Source Datareplicator definitions

Name of source Datareplicator definition	Definition filename	Required/optional	Description
Extraction system definition	\$HDEPATH/hdeenv	Required	Defines information about the source Datareplicator's operating environment. The defined information includes the source and target Datareplicator identifiers.
Extraction environment definition	Any name	Required	Defines information about the extraction processing environment. The defined information includes the names and size of the extraction information queue files.
Transmission environment definition	Any name	Required	Defines information about a transmission processing environment. The defined information includes the service and host names for communications.
Extraction definition	Any name	Required	Defines information about extraction and transmission processing. The defined information includes the correspondence between a table subject to extraction processing and its update information and the target identifiers for the update information.
Source HiRDB definition	The following HiRDB definition files: <ul style="list-style-type: none"> • System common definition file • Unit control information definition file 	Required	Defines information needed to use the source Datareplicator at the source HiRDB. The defined information includes the data linkage start method and the name of the directory used by the source Datareplicator.
Duplexing definition	File name specified in the <code>file_dupenv</code> operand in the extraction system definition file	Optional	Defines information needed to duplex the definition files that are used at the source system. The files that can be duplexed include the extraction master status file, extraction server status file, extraction information queue file, and data linkage file.

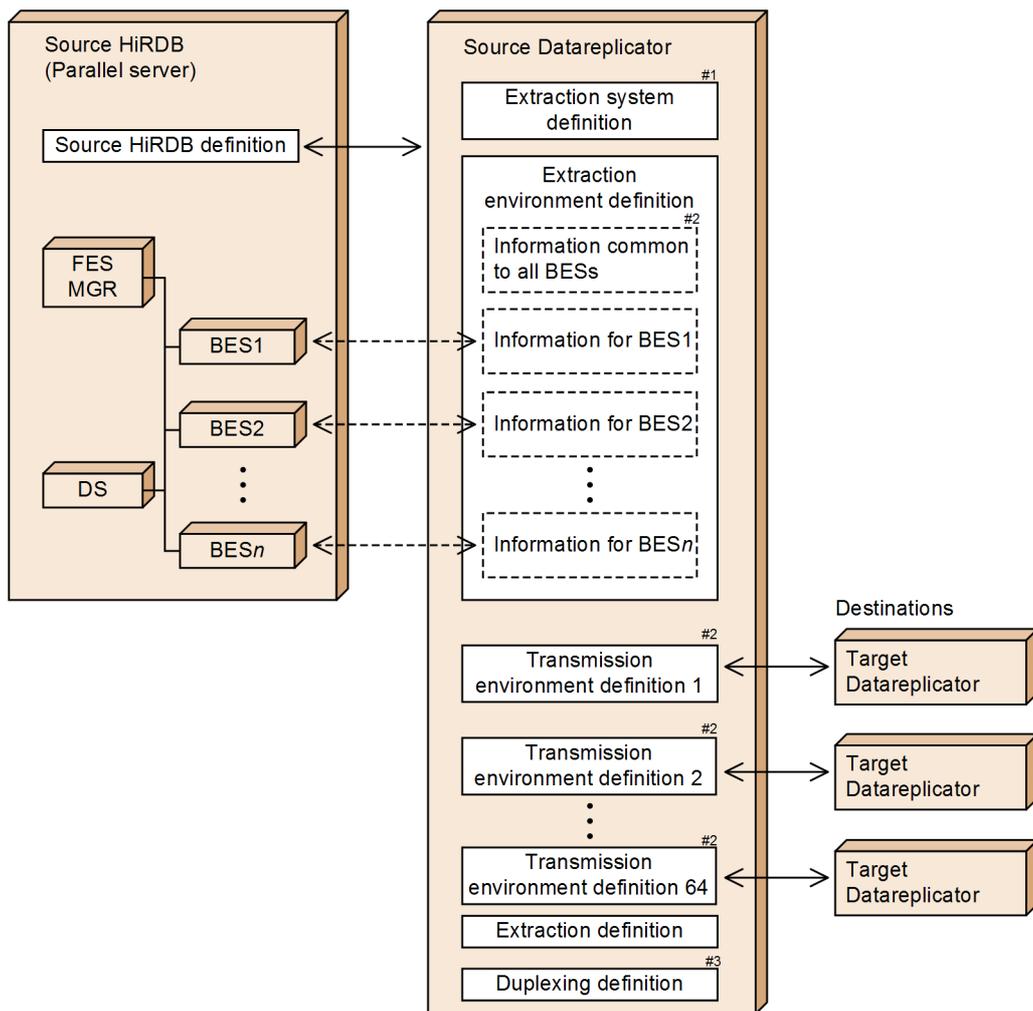
Table 5-2: Target Datareplicator definitions

Name of target Datareplicator definition	Definition filename	Required/optional	Description
Import system definition	\$HDSPATH/hdsenv	Required	Defines information about the target Datareplicator's operating environment. The defined information includes the communications protocol, source system's data linkage identifier, and so on.
Import environment definition	Any name	Required	Defines information about an import processing environment. The defined information includes the import information queue filenames, import status filename, and import method.
Import definition	Any name	Optional	Defines information about import processing. The defined information includes the correspondence between update information and the table subject to import processing, the target front-end server, key range partitioning conditions, hash partitioning conditions, and UOC routine names.
Update information definition	Any name	Optional	Defines update information when the source database is PDM2 E2 and SAM files are used for data linkage.
Duplexing definition	File name specified in the <code>file_dupenv</code> operand in the import system definition file	Optional	Defines information needed to duplex the definition files that are used at the target system. The files that can be duplexed include the extraction master status file, extraction server status file, extraction information queue file, and data linkage file.

(1) Relationship between the source Datareplicator definitions and the data linkage system

The following figure shows the relationship between the source Datareplicator definitions and the data linkage system.

Figure 5-2: Relationship between the source Datareplicator definitions and the data linkage system



Legend:

MGR : HiRDB system manager FES : HiRDB front-end server
 BES : HiRDB back-end server DS : HiRDB dictionary server

#1

If the source HiRDB is a parallel server and different information is needed in the environment variables for each server machine, you can specify the operands for the environment variables individually for each server machine.

#2

You can specify information common to all back-end servers for the source HiRDB or you can specify information only for specific back-end servers as needed.

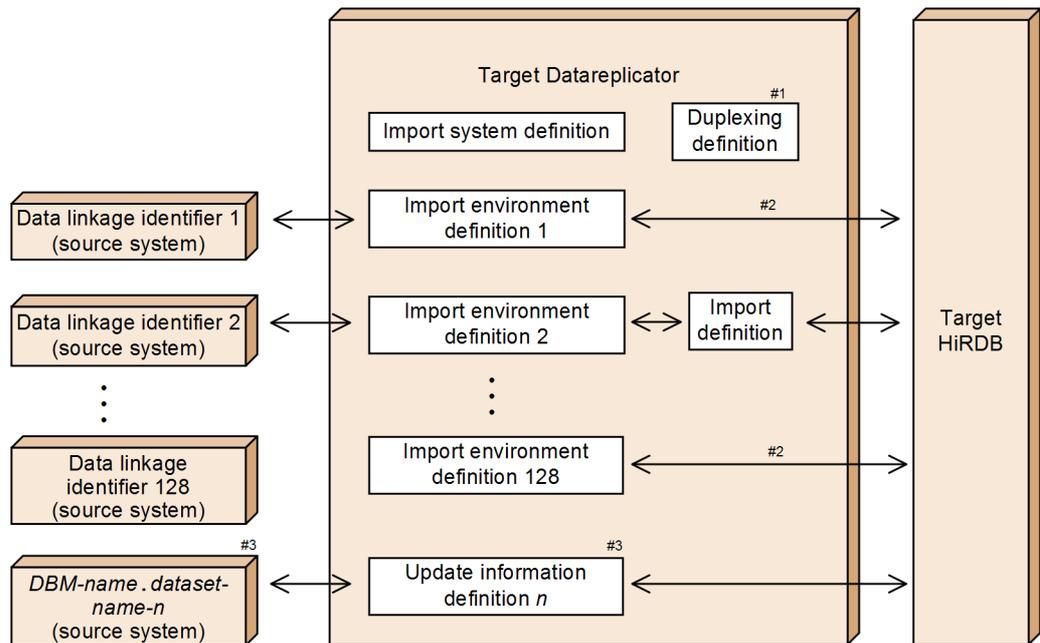
#3

The duplexing definition is required to use the file duplexing feature.

(2) Relationship between the target Datareplicator definitions and the data linkage system

The following figure shows the relationship between the target Datareplicator definitions and the data linkage system.

Figure 5-3: Relationship between the target Datareplicator definitions and the data linkage system



#1

The duplexing definition is required to use the file duplexing feature.

#2

You can omit the import definition if the source and target tables have exactly the same table format, table name, and column names.

#3

The update information definition is required for PDMII E2 when the source system uses SAM files.

5.1.2 Definition rules

Datereplicator definitions are specified in three formats:

`set` format

The `set` format applies to the extraction system definition, extraction environment definition, transmission environment definition, import system definition, and import environment definition.

Syntax format

The syntax format applies to the extraction definition, import definition, and update information definition.

File-specific format

Duplexing definition

If the source HiRDB is a parallel server and different information is specified in HiRDB's environment variables (such as `PDDIR`, `PDCONFPATH`, and `SHLIB_PATH`) for each server machine, you can specify the operands for the environment variables individually for each such server machine.

If the source HiRDB is a parallel server, you can define an extraction environment definition and a transmission environment definition, which are both source Datereplicator definitions, for each back-end server.

(1) *Definition rules for the set format*

- Specify each definition statement as 1-80 single-byte characters per line. If a definition statement exceeds 80 characters, enter the backslash (\) continuation symbol immediately before the linefeed code, and then continue the specification on the next line. The continuation line must begin in column 1.
- A comment begins with the hash mark (#). Datereplicator regards any information between a hash mark (#) and the end of the line as a comment. A comment cannot continue onto a subsequent line.
- There are no restrictions on the specification order of operands.
- If you specify the same operand more than once, the last specification in the definition file is effective.
- A parameter name without a parameter value will result in an invalid parameter value error.

(2) Definition rules for the syntax format

- A comment must begin with /* and end with */.
- A comment can continue onto subsequent lines; however, each set of delimiter characters (/* and */) must be entirely on one line.
- To specify a single quotation mark (') in a character string literal, specify two consecutive single quotation marks.
- To use a Datareplicator reserved word as an identifier, enclose it in double quotation marks ("). For details about the Datareplicator reserved words, see Appendix B. *Datareplicator Reserved Words*.
- Specify a semicolon (;) at the end of the final definition statement.

(3) Definition rules for file-specific format

For details about the format of a duplexing definition, see 5.7 *Duplexing definition (source)* or 5.12 *Duplexing definition (target)*.

(4) Definitions for individual server machines (extraction system definition)

If the source HiRDB is a parallel server and different information is specified in HiRDB's environment variables (such as PDDIR, PDCONFPATH, and SHLIB_PATH) for each server machine, you can specify the operands for the environment variables individually for each such server machine. If you omit the definition for any specific server machine, the source Datareplicator at that server machine will use the environment variables defined for the user environment in which the source Datareplicator's commands are executed. This section explains the format and rules for creating definitions for individual server machines.

For each such definition, the source Datareplicator creates a temporary work file named *definition-filename_tmp* under HDEPATH during analysis of definition files.

(a) Format for definitions for individual server machines

The following figure shows the format for definitions for individual server machines.

whose values are to be common to all back-end servers, and specify in an individual definition section the operands whose values are to be specific to the particular back-end server.

- If all back-end servers use the same definitions, specify only the common definition section.
- Specify the common definition section at the beginning of the definition file.
- You can omit the string `commondef`. Datareplicator assumes the beginning of the definition file up to the beginning of the first `besdef (server-name)` as the common definition section, regardless of whether the string `commondef` is specified. If it detects no `besdef (server-name)` string, Datareplicator assumes that the entire information is the common definition section.
- You can define all operands in both the common definition section and individual definition sections. If both definition sections are specified for a particular back-end server, the information in the individual definition section takes precedence. If a back-end server has no individual definition section, the information in the common definition section is effective for that back-end server. If an operand is not defined in the individual definition section for a back-end server but is defined in the common definition section, the specification in the common definition section is effective for that back-end server.
- If the same `besdef (server-name)` string is specified more than once, the first one specified is effective.
- If the server specified in `besdef (server-name)` does not belong to the source HiRDB, an error occurs during definition analysis.
- Specify `commondef` and `besdef (server-name)` using single-byte lowercase letters.
- Specify in `besdef (server-name)` the server name of the back-end server being defined. Do not include any spaces in `besdef (server-name)`.

5.2 Extraction system definition

The extraction system definition specifies information about the entire source Datareplicator's operating environment.

5.2.1 Format

Items defined in the common definition section only

```

set hdeid=source-Datareplicator-identifier
[ set extdef="extraction-environment-definition-filename" ]
[ set sendcontrol=nodemst|sendmst ]
set sendid01=target-identifier
[... [ set sendid64=target-identifier ] ]#1
[ set senddef01="transmission-environment-definition-filename" ]
[... [ set senddef64="transmission-environment-definition-filename" ] ]#1
set sendid0001=target-identifier
[... [ set sendid4096=target-identifier ] ]
[ set senddef0001=transmission-environment-definition-filename ]
[... [ set senddef4096=transmission-environment-definition-filename ] ]
[ set errfile_unique=true|false ]
[ set errfilesz=maximum-error-information-file-size ]
[ set syslogout=true|false ]
[ set syslog_message_suppress=message-number[ ,message-number] ... ]
[ set dblocale={ sjis|euc|utf-8 } ]
[ set msglocale={ english|sjis-japanese|euc-japanese } ]
[ set watchintvl=error-monitoring-interval ]
[ set cmwaittime=communication-wait-time ]
[ set mstservice=master-to-node-master-communication-service-name ]#2
[ set extinfunum=maximum-update-information-names-count ]
[ set syncterm=true|false ]
[ set termlevel={ normal|plan|both } ]
[ set info_message_out=nosuppress|suppress ]
[ set except_suppress=message-number[ ,message-number] ... ]
[ set int_trc_lvl=activity-trace-collection-level[ ,activity-trace-collection-range] ]
[ set int_trc_filesz=activity-trace-file-size ]
[ set int_trc_rintvl=activity-trace-information-collection-interval ]
[ set sendprocnum=maximum-transmission-processes-to-be-started ]
[ set smt_sendintvl=transmission-master-process-transmission-interval ]
[ set smt_sendintvl_scale=minute|second ]
[ set smt_editbufsize=update-information-editing-buffer-size ]
[ set smt_readbufnum=I/O-buffers-count-for-reading-update-information ]
[ set smt_queue_read_wait_interval=transmission-process's-extraction-information-queue-file-read-interval ]
[ set file_dupenv=duplexing-definition-file-name ]
[ set nodecontrol=unit|server ]
[ set node_connection_accept=true|false ]

```

```
[ set connection_accept_hostname=
extraction-node-master-process-connection-request-accepting-host-name ]
[ set connection_accept_service=
extraction-node-master-process-connection-request-accepting-service-name ]
[ set connection_accept_waittime=
extraction-node-master-process-connection-request-wait-time ]
[ set connection_retry_time=
extraction-node-master-process-reconnection-processing-time ]
[ set node_syslogout=true|false ]
[ set send_counter_reset=true|false ]
[ set hirdb_audit_trail = all | none ]
[ set resource_chk_err = continue | stop ]
[ set recover_info_send = true|false ]
[ set recover_info_send_interval =
recovery-information-transmission-interval ]
[ set cm_errno_check = true|false ]
```

Items defined in individual definition sections only

```
[ set node_pddir=source-HiRDB's-PDDIR-environment-variable-value ]
[ set node_pdconffpath=HiRDB's-PDCONFPATH-environment-variable-value ]
[ set node_shlibpath=HiRDB's-SHLIB_PATH-environment-variable-value ]
```

#1: If you are using Windows Datareplicator, you must not specify `sendid64` or `senddef64`, because Datareplicator allows only up to 63 target identifiers and transmission environment definition files; an error will result if you specify either of these operands.

#2: In the case of Windows Datareplicator, the master-to-node master communication service name is always `hdenmserv`; specify this name in the `services` file after you have installed Datareplicator. Do not specify the `mstservice` operand; it will be ignored if specified.

5.2.2 Modifying defined information

To modify defined information:

1. Terminate the source Datareplicator.

To execute an initial start subsequently, terminate the source Datareplicator normally. If the source Datareplicator has terminated abnormally, start it normally, and then perform the procedure to modify the defined information.
2. Use a text editor to modify the defined information.
3. Start the system in a start mode that will apply the modified information, as shown in the following table.

Table 5-3: Start mode that applies the modified extraction system definition

Operand name	Start mode that applies the modified information		
	Initial start ^{#1}	Partial initial start ^{#1}	Normal start ^{#1}
hdeid	Y	--	--
extdef	Y	--	--
sendcontrol	Y	--	--
sendidx ^{#2}	Y	Y	--
senddefxx ^{#2}	Y	Y	--
sendidxxx ^{#3}	Y	Y	--
senddefxxx ^{#3}	Y	Y	--
errfile_unique	Y	--	--
errfilesz	Y	Y	Y
syslogout	Y	Y	Y
syslog_message_suppress	Y	Y	Y
dblocale	Y	--	--
msglocale	Y	Y	Y
watchintvl	Y	Y	Y
cmwaittime	Y	Y	Y
mstservice	Y	Y	Y
extinforum	Y	--	--
syncterm	Y	Y	Y
termlevel	Y	Y	Y
info_message_out	Y	Y	Y
except_suppress	Y	Y	Y
int_trc_lvl	Y	Y	Y
int_trc_filesz	Y	Y	Y
int_trc_rintvl	Y	Y	Y

Operand name	Start mode that applies the modified information		
	Initial start ^{#1}	Partial initial start ^{#1}	Normal start ^{#1}
sendprocnum	Y	Y	Y
smt_sendintvl	Y	Y	Y
smt_sendintvl_scale	Y	Y	Y
smt_editbufsize	Y	Y	Y
smt_readbufnum	Y	Y	Y
smt_queue_read_wait_interval	Y	Y	Y
file_dupenv	Y	--	--
nodecontrol	Y	--	--
node_connection_accept	Y	Y	Y
connection_accept_hostname	Y	Y	Y
connection_accept_service	Y	Y	Y
connection_accept_waittime	Y	Y	Y
connection_retry_time	Y	Y	Y
node_syslogout	Y	Y	Y
send_counter_reset	Y	Y	Y
hirdb_audit_trail	Y	Y	Y
resource_chk_err	Y	Y	Y
recover_info_send	Y	--	--
recover_info_send_interval	Y	--	--
cm_errno_check	Y	Y	Y
node_pddir	Y	--	--
node_pdconfpath	Y	--	--
node_shlibpath	Y	--	--
node_host	Y	Y	Y

Legend:

Y: The start mode denoted by this column applies the operand's modified information. If the start modes of multiple columns apply a particular operand's modified information, any one of those start modes can be used to apply the modified information.

--: Not applicable

#1

This is the start mode used to start the source Datareplicator.

#2

xx is a value in the range from 01 to 64. If you are executing a normal start, do not change any of the `sendid01` to `sendid64` or `senddef01` to `senddef64` operands.

#3

xxx is a value in the range from 0001 to 4096. If you are executing a normal start, do not change any of the `sendid0001` to `sendid4096` or `senddef0001` to `senddef4096` operands.

5.2.3 Explanation of the operands

- **hdeid=source-Datareplicator-identifier**

~ <hexadecimal> ((00-FF))

Specify the identifier to be used to identify this source system. If the source HiRDB is a parallel server, one source system includes the source Datareplicator under the system manager and the source Datareplicators under all back-end servers. Thus, if the source HiRDB is a parallel server, you need one source Datareplicator identifier per source system.

- **extdef="extraction-environment-definition-filename"**

~ <[*pathname*/]*filename* of 1-64 bytes> <<extraction environment definition used when all operands are omitted>>

Specify the filename of the extraction environment definition that defines the operating environment for extraction processing, as an absolute or relative pathname. If you specify a relative pathname, Datareplicator assumes `$HDEPATH/relative-pathname` as the absolute pathname. In this case, specify the pathname so that the final absolute pathname will not exceed 125 characters. If you omit the `extdef` operand, Datareplicator assumes the extraction environment definition that is used when all operands are omitted.

- **sendcontrol=nodemst|sendmst**

Specify the processing method to be used to send extracted data.

`nodemst`

Start a transmission process for each destination and execute transmission processing for all destinations in parallel. Use the `sendidxx` and `senddefxx` operands to specify a maximum of 64 destinations.

We recommend that you specify `nodemst` for an application that has only a few destinations with a high volume of traffic.

`sendmst`

Use the transmission master process to control the number of transmission processes. The transmission master process schedules transmission processing for the number of transmission processes specified in the `sendprocnum` operand. Use the `sendidxxx` and `senddefxxx` operands to specify a maximum of 4,096 destinations.

We recommend that you specify `sendmst` for an application that has many destinations with a low volume of traffic.

■ **`sendid01=target-identifier[...[sendid64=target-identifier]]`**

~ <symbolic name of 1-8 characters or **>

Specify the names that identify the destinations of update information (target identifiers). These target identifiers are used to identify the target systems subject to processing when transmission statements or commands are specified in the extraction definition. These target identifiers are also displayed in messages.

When you specify `nodemst` in the `sendcontrol` operand or you omit the `sendcontrol` operand, you must specify the `sendidxx` operand. This operand is effective only when `nodemst` is specified in the `sendcontrol` operand or the `sendcontrol` operand is omitted.

You must specify the `sendid01` to `sendid64` operands consecutively in ascending order beginning with `sendid01`. If they are not consecutive or in ascending order, Datareplicator uses only those operands that are specified consecutively in ascending order from the beginning.

When you specify `**` in place of a target identifier, Datareplicator assumes an absent number, and the corresponding `senddefxx` operand is ignored (transmission to a target will not occur). You can specify `**` in place of any number of target identifiers; if you specify `**` for all target identifiers, Datareplicator assumes that no target identifiers at all are specified.

In the case of Windows Datareplicator, you can specify a maximum of 63 target identifiers corresponding to transmission environment definition files; an error will result if `sendid64` is specified.

- **senddef01="transmission-environment-definition-filename" [... [senddef64="transmission-environment-definition-filename"]]**

~ <[*pathname/*]*filename* of 1-64 bytes> <<transmission environment definition used when all operands are omitted>>

Specify the filenames of the transmission environment definitions that define operating environments for transmission processing, as absolute or relative pathnames. If you specify a relative pathname, Datareplicator assumes `$HDEPATH/relative-pathname` as the absolute pathname. In this case, specify the pathname so that the final absolute pathname will not exceed 125 characters.

This operand is effective only when `nodemst` is specified in the `sendcontrol` operand or the `sendcontrol` operand is omitted. Each specified filename must be unique in the source system.

You must specify the `senddef01` to `senddef64` operands so that they correspond to the `sendid01` to `sendid64` operands. If there is no `sendidnn` operand corresponding to a `senddefxx` operand, that `senddefxx` operand will be ignored. On the other hand, if there is a `sendidnn` operand but no corresponding `senddefxx` operand, Datareplicator assumes the transmission environment definition that is used when all operands are omitted.

In the case of Windows Datareplicator, you can specify a maximum of 63 transmission environment definition files corresponding to target identifiers; an error will result if `senddef64` is specified.

- **sendid0001=target-identifier [... [sendid4096=target-identifier]]**

~ <symbolic name of 1-8 characters>

Specify the names that identify the destinations of update information (target identifiers). These target identifiers are used to identify the target systems subject to processing when transmission statements or commands are specified in the extraction definition. These target identifiers are also displayed in messages.

When you specify `sendmst` in the `sendcontrol` operand, you must specify the `sendidxxxx` operand. This operand is effective only when `sendmst` is specified in the `sendcontrol` operand.

You must specify the `sendid0001` to `sendid4096` operands consecutively in ascending order beginning with `sendid0001`. If they are not consecutive or in ascending order, Datareplicator uses only those operands that are specified consecutively in ascending order from the beginning.

When you specify `**` in place of a target identifier, Datareplicator assumes an absent number, and the corresponding `senddefxxxx` operand is ignored (transmission to a target will not occur). You can specify `**` in place of any number of target identifiers; if you specify `**` for all target identifiers, Datareplicator assumes that no target identifiers at all are specified.

- **senddef0001="transmission-environment-definition-filename" [... [senddef4096="transmission-environment-definition-filename"]]**

~ <[*pathname*/]*filename* of 1-125 bytes> <<transmission environment definition used when all operands are omitted>>

Specify the filenames of the transmission environment definitions that define operating environments for transmission processing, as absolute or relative pathnames. If you specify a relative pathname, Datareplicator assumes \$HDEPATH/*relative-pathname* as the absolute pathname. In this case, specify the pathname so that the final absolute pathname will not exceed 125 characters.

This operand is effective only when `sendmst` is specified in the `sendcontrol` operand. Each specified filename must be unique in the source system.

You must specify the `senddef0001` to `senddef4096` operands so that they correspond to the `senddid0001` to `senddid4096` operands. If there is no `sendidxxxx` operand corresponding to a `senddefxxxx` operand, that `sendidxxxx` operand will be ignored. On the other hand, if there is a `sendidxxxx` operand but no corresponding `senddefxxxx` operand, Datareplicator assumes the transmission environment definition that is used when all operands are omitted.

- **errfile_unique=true|false**

In the case of a HiRDB/Parallel Server system that employs the mutual system switchover configuration, specify whether the names of the extraction node master error information files are to be expanded so that they will be unique throughout the source Datareplicator. This specification also applies to the names of the extraction node master trace files.

true

Datareplicator is to add the applicable node's host name to the error information filenames and activity trace filenames so that the names are unique in the source Datareplicator.

- Extraction node master error information filenames:
`errfile1_host-name`, `errfile2_host-name`
- Extraction node master trace filenames: `exttrc_host-name.trc1`,
`exttrc_host-name.trc2`

false

Datareplicator is to use the same names for the error information files (`errfile1` and `errfile2`) and the extraction node master trace files (`exttrc.trc1` and `exttrc.trc2`) at all nodes of the HiRDB/Parallel Server.

Specify `true` in the `errfile_unique` operand only when Datareplicator is used in a HiRDB/Parallel Server system that employs the mutual system

switchover configuration. Otherwise, specify `false` or omit the `errfile_unique` operand.

■ **errfilesz=maximum-error-information-file-size**

~ <unsigned integer> ((1-32767)) <<16>> (KB)

Specify a maximum size for the extraction master error information files and extraction node master error information files. For details about the sizes for the extraction master error information files and extraction node master error information files, see *4.6.8 Designing the source Datareplicator's resources*.

■ **syslogout=true|false**

Specify whether the information in the extraction master error information files is to be output also to the syslog file. If you are using JP1, you must output this information to the syslog file to implement automatic operation.

`true`

Output the information in the extraction master error information files also to the syslog file.

`false`

Do not output the information in the extraction master error information files to the syslog file.

Note that the `KFRB00501-I` and `KFRB00505-I` messages are still output to the syslog file even when `false` is specified.

■ **syslog_message_suppress=message-number[,message-number] ...**

Specify the numbers of the messages whose output to the syslog file (event log for Windows) is to be suppressed.

- Specify a maximum of 64 message numbers.
- If you specify the same message number more than once, only the first specification of the number is effective.
- Datareplicator does not output any of the messages specified in this operand, regardless of a message's significance level (E, W, I, or Q).
- Messages that are output only to the syslog file (or event log) are always output regardless of whether you specify this operand.

■ **dblocale=sjis|euc|utf-8**

Specify the character code system supported by the source Datareplicator. Datareplicator assumes that the extraction definitions are coded in the code system specified in this operand when it analyzes extraction definitions. Specify the character code system used at the source HiRDB.

`sjis`

Use the JIS8/Shift JIS code system.

`euc`

Uses the EUC code system.

`utf-8`

Uses the utf-8 code system.

The default value for this operand depends on the OS being used, as shown in the following table:

OS	Default value
HP-UX	sjis
AIX	
Windows	
Solaris	euc
Linux	

In the case of Windows Datareplicator, you must specify either `sjis` or `utf-8`, because the source Datareplicator supports only the JIS8/Shift JIS and Unicode systems. Specification of any other value in the `dblocate` operand will result in an error.

■ **msglocale={ english|sjis-japanese|euc-japanese }**

Specify the character code set to be used for messages issued by the source Datareplicator.

For Windows Datareplicator, specify either `english` or `sjis-japanese`; specifying `euc-japanese` will result in an error.

`english`

Output messages in English.

`sjis-japanese`

Output messages in Japanese using the JIS8/Shift JIS code system.

`euc-japanese`

Output messages in Japanese using the EUC code system.

■ **watchintvl=error-monitoring-interval**

~ <unsigned integer> ((10-32767)) <<60>> (seconds)

Specify the interval at which the extraction master process is to monitor extraction and transmission processing for errors, or the interval at which a delay between extraction and transmission processes is to be monitored.

■ **cmwaittime=communication-wait-time**

~ <unsigned integer> ((10-32767)) <<60>> (seconds)

Specify the wait time for communication to occur between the extraction master process and the extraction node master process.

■ **mstservice=master-to-node-master-communication-service-name**

~ <identifier of 1-64 characters> <<hdemaster>>

Specify the service name to be used to add to the `services` file a communication entry between the source Datareplicator's extraction master process and extraction node master process.

For Windows Datareplicator, the master-to-node master communication service name is always `hdenmserv`, and this name is specified in the `services` file after Datareplicator installation is completed. Therefore, you need not specify the `mstservice` operand in this case; Datareplicator will ignore this operand if it is specified for Windows Datareplicator.

■ **extinfo num=maximum-update-information-names-count**

~ <unsigned integer> <<50>> ((1-4096))

Specify the number of update information names that can be specified in the extraction definition. If this value is less than the number of update information names actually specified in the extraction definition, an error occurs during execution of the `hdeprep` command.

■ **syncterm=true|false**

Specify whether the source Datareplicator is to terminate normally automatically when the source HiRDB terminates normally.

In the case of a system switchover configuration, all update information might not always be sent to the target system even when `true` is specified in this operand, because the order of the system switchover timing and the source Datareplicator's termination timing cannot be predicted. Therefore, do not specify `true` in this operand in the case of a system switchover configuration. If you perform planned system switchover after all update information has been sent to the target system, do so after using the `hdestate` command to check the status.

`true`

Automatically terminate the source Datareplicator normally when the source HiRDB terminates normally. All the following conditions must be satisfied:

- The source HiRDB's normal termination is detected.
- No transactions have occurred at the source HiRDB since detection of the source HiRDB's normal termination.
- The extraction process has completed extracting all update information from the system log into the extraction information queue file.
- The transmission process has completed sending all update information from the extraction information queue file to the target system.

`false`

Do not terminate the source Datareplicator normally when the source HiRDB terminates normally.

- If `true` is specified, the source Datareplicator terminates itself after extracting all system log information from the system log file and sending all the extracted update information subject to transmission to the applicable target systems.

A transmission process must have already started before the extraction process detects the source HiRDB's normal termination. Otherwise, the source Datareplicator terminates normally when the extraction process detects the source HiRDB's normal termination, in which case some update information might remain in the extraction information queue file. If this happens, Datareplicator sends the remaining update information from the extraction information queue file to the target systems the next time the transmission process is started by the `hdestart` or `hdestart -s` command.

- If you will be using the facility for recovering the extraction information queue file, specify `false`. When `true` is specified, the facility for recovering the extraction information queue file cannot be used.

■ **termlevel={ normal|plan|both }**

When the source Datareplicator is specified to terminate normally automatically when the source HiRDB terminates, specify the HiRDB termination method that is to be subject to this synchronized termination. This operand is applicable only when `true` is specified in the `syncterm` operand.

`normal`

Terminate the source Datareplicator normally only when the HiRDB terminates normally.

`plan`

Terminate the source Datareplicator normally only when planned termination of the HiRDB occurs.

`both`

Terminate the source Datareplicator normally both when the HiRDB terminates normally and when planned termination of the HiRDB occurs.

■ **`info_message_out=nosuppress|suppress`**

Specify whether output of information-only messages to the syslog file (event log for Windows) and error information files is to be suppressed.

`nosuppress`

Do not suppress output of information-only messages.

`suppress`

Suppress output of information-only messages. The following message numbers are subject to this operand's specification:

00551, 00552, 00553, 00554, 00555, 00557, 00581, 00583, 02031,
02032, 02033, 02034, 02036, 02037, 05001, 05002, 05008, 05012,
05013, 05018

■ **`except_suppress=message-number[,message-number] ...`**

~ <5-digit unsigned integer>

From among the messages whose output is suppressed because `info_message_out=suppress` is specified, specify the numbers of the messages whose output suppression is to be cancelled and which are to be output to the syslog and error information files. You can specify a maximum of 63 message numbers.

Datareplicator ignores specification of a message number if it is not subject to output suppression. If you specify the same message number more than once, only the first specification is effective.

This operand is applicable only when `suppress` is specified in the `info_message_out` operand.

■ **`int_trc_lvl=activity-trace-collection-level[,activity-trace-collection-range]`**

To change the items to be collected in activity trace files (extraction master trace files and extraction node master trace files), specify the appropriate value shown in the table below. If you omit this operand, Datareplicator collects only the information common to all facilities (the minimum requirement).

If you specify `na` for the activity trace collection level, Datareplicator will not collect any activity trace information and will ignore any value specified for the activity trace collection range (however, Datareplicator still checks for syntax and range errors).

- Value for the activity trace collection level

Value	Information to be collected		
	Common information	Overview of performance	Details about performance
p1	Y	Y	--
p2	Y	Y	Y
na	--	--	--
int_trc_lvl operand omitted	Y	--	--

Y: Collected.

--: Not collected.

Common information: Collective name for the general checkpoint information that indicates a point of change in start/termination information, error information, or process level.

Note:

When you specify p2, you must provide sufficient space for the activity trace files. Otherwise, important information might be deleted by the wraparound feature; in addition, trace collection might result in a large overhead workload.

- Value for the activity trace collection range

Value	Information to be collected		
	MST (control facility)	CAP (extraction facility)	SND (transmission facility)
c1	Y	Y	--
c2	Y	--	Y
c3	Y	--	--
nc	Y	#	#
int_trc_lvl operand or <i>activity-trace-collection-range</i> omitted	Y	Y	Y

Y: Collected.

--: Not collected.

#: Even if you specify `nc` in the extraction system definition, you can still collect activity trace information individually by specifying the `int_trc_getv` operand in the extraction environment definition and the transmission environment definition.

Guidelines for specifying the `int_trc_lvl` operand:

We recommend that you specify the `int_trc_lvl` operand as follows:

1. Production run

For actual operations, we recommend that you omit the `int_trc_lvl` operand. If you cannot obtain reasonable performance with specification of the operand omitted, specify collection of activity trace information temporarily so that you can evaluate performance. In this case, specify `p1` or `p2` in the first parameter of the `int_trc_lvl` operand, `nc` in the second parameter, and the `int_trc_getl` or `int_trc_getv` operand in the specific facility (the one exhibiting low performance). If you specify `p2` in the first parameter of the `int_trc_lvl` operand, specify a sufficient value in the `int_trc_filesz` operand.

2. Test run

At the test stage, we recommend that you specify `p1` in the first parameter of the `int_trc_lvl` operand and omit the second parameter. This enables you to execute the `hdstrcredit` command to view activity trace information in the event of a problem with performance. Use the command's execution results to tune the HiRDB table definitions and the number of import groups.

If you need more detailed information (such as the unit price of SQL executions), change the second parameter to `p2` in the `int_trc_lvl` operand. When you specify `p2`, specify a sufficient value in the `int_trc_filesz` operand (at least 1MB).

■ **int_trc_filesz=activity-trace-file-size**

~ <unsigned integer> ((32-1048576)) <<128>> (KB)

Specify the maximum storage size for an activity trace file (extraction master trace file or extraction node master trace file). Datareplicator ignores this operand when `na` is specified in the `int_trc_lvl` operand.

We recommend that you specify a value that is a multiple of 32. Otherwise, Datareplicator rounds up the specified value to the nearest multiple of 32KB and uses that value as the maximum size for an activity trace file.

In the source system, Datareplicator creates the activity trace files under the following names in the `$HDEPATH` directory (Datareplicator uses dual files with swapping and wrapping):

Extraction master trace files

`$HDEPATH/msttrc.trc1` and `$HDEPATH/msttrc.trc2`

Extraction node master trace files

`$HDEPATH/exttrc.trc1` and `$HDEPATH/exttrc.trc2`

If you specify `true` in the `errfile_unique` operand, `_host-name` is added to the filename (that is, the filenames become `exttrc_host-name.trc1` and `exttrc_host-name.trc2`).

■ **int_trc_rintvl=activity-trace-information-collection-interval**

~ <unsigned integer> ((5-30000)) <<50>> (milliseconds)

Specify the interval at which activity trace information is to be collected. If `na` is specified in the `int_trc_lvl` operand, Datareplicator ignores this operand if it is specified.

You can minimize the amount of missed activity trace information by specifying a short activity trace information collection interval (specify a small value), but the number of monitorings per second increases, resulting in a higher CPU utilization factor. Specify a small activity trace information collection interval in the following cases:

- There are missing entries in the activity trace information.
- The specified activity trace collection level indicates a high level of collection information.
- The frequency of Datareplicator activity is high (there is a large amount of extraction data).

On the other hand, specify a large activity trace information collection interval in the following case:

- The frequency of Datareplicator activity is low (there is a small amount of extraction data) and a reduction of CPU utilization by Datareplicator is desired.

■ **sendprocnum=maximum-transmission-processes-to-be-started**

UNIX Datareplicator: ~ ((1-4096)) <<1>>

Windows Datareplicator: ~ ((1-63)) <<1>>

Specify the maximum number of transmission processes that can be started by the transmission master process. This operand is applicable when `sendmst` is specified in the `sendcontrol` operand.

■ **smt_sendintvl=transmission-master-process-transmission-interval**

~ ((0-1440)) <<5>>

Specify the transmission master process's transmission interval. Use the

`smt_sendintvl_scale` operand to specify the units of the transmission interval. This operand is applicable when `sendmst` is specified in the `sendcontrol` operand. If you specify 0, the transmission master process will send update information in units of transactions. If you want to reduce the time leading up to import processing, start tuning from one second. Specifying 0 might slow down the transmission processing.

When `sendmst` is specified in the `sendcontrol` operand, Datareplicator ignores the `sendintvl` operand in the transmission environment definition and uses the value of this operand to execute transmission processing.

■ **`smt_sendintvl_scale=minute|second`**

Specify the units of the `smt_sendintvl` operand's value.

`minute`

Use minutes for the `smt_sendintvl` operand's value.

`second`

Use seconds for the `smt_sendintvl` operand's value.

This operand is applicable when the `smt_sendintvl` operand is specified.

■ **`smt_editbufsize=update-information-editing-buffer-size`**

~ ((1-2097151)) <<300>> (KB)

Specify the buffer size needed by the transmission process to edit the format of update information so that the target system can receive it when the transmission process is controlled by the transmission master process.

This operand is applicable when `sendmst` is specified in the `sendcontrol` operand, in which case Datareplicator ignores the `editbufsize` operand in the transmission environment definition and uses the value of this operand to execute transmission processing. You can reduce the number of communications by specifying a large value in this operand. We recommend that you specify a value that is greater than the amount of data (amount of update information) that is written to the extraction information queue file within the amount of time specified in the `smt_sendintvl` operand.

To determine the amount of data that is written to the extraction information queue file, execute periodically the `hdestate` command and measure the amount of increase in the offset information indicated by the Queue write position.

■ **`smt_readbufnum=I/O-buffers-count-for-reading-update-information`**

~ ((1-255)) <<1>>

Specify the number of I/O buffers to be used by the transmission master process to read update information from the extraction information queue file when the

transmission process is controlled by the transmission master process.

This operand is applicable when `sendmst` is specified in the `sendcontrol` operand, in which case Datareplicator ignores the `readbufnum` operand in the transmission environment definition and uses the value of this operand to execute transmission processing.

■ **smt_queue_read_wait_interval=transmission-process's-extraction-information-queue-file-read-interval**

~ <unsigned integer> ((100-60000)) <<2000>> (milliseconds)

Specify the interval after which the next read operation is to be started once the end of the extraction information queue file is detected when the transmission is controlled by the transmission master process.

By specifying a small value in this operand, you can improve the spontaneity of transmission processing, because there is only a brief wait after the end of the extraction information queue file is detected. However, this might have an adverse effect on performance, because the CPU utilization factor increases. You need to take into account the frequency of update processing and CPU performance when specifying this value.

This operand is applicable when `sendmst` is specified in the `sendcontrol` operand, in which case Datareplicator ignores the `queue_read_wait_interval` operand in the transmission environment definition and uses the value of this operand to execute transmission processing.

■ **file_dupenv=duplexing-definition-file-name**

~ <filename of 1 to 125 bytes>

Specify the absolute or relative path name of the duplexing definition file. If a relative path is specified, the system assumes that the path is relative to the source Datareplicator directory. If this operand is omitted, Datareplicator assumes that the duplexing function is not used.

■ **nodecontrol= unit|server**

Specify how to control the extraction node master process.

`unit`

Start the extraction node master process for each unit of the HiRDB subject to extraction processing.

`server`

Start the extraction node master process for each back-end server of the HiRDB subject to extraction processing.

You must specify `server` if the HiRDB subject to extraction processing uses

a standby-less system switchover (effects distributed) configuration.

If you use the `hdestart` command to start the extraction master process, the HiRDB subject to extraction processing must be running so that start host information for the extraction node master process can be acquired from the HiRDB subject to extraction processing (if you use the `hdestart_n` command to start the extraction master process, there is no need for the HiRDB subject to extraction processing to be running).

The following table shows the resources whose name and creation unit depend on this operand's value:

Resource type	nodecontrol=unit specified		nodecontrol=server specified#	
	Name	Creation unit	Name	Creation unit
Extraction node master error information file	errfile1, errfile2	2 for each node	errfile1_server-name, errfile2_server-name	2 for each server
Extraction node master activity trace file	exttrc.trc1, exttrc.trc2		exttrc_server-name.trc1, exttrc_server-name.trc2	

#: The system ignores the `errfile_unique` operand in the extraction system definition and creates resources using the names shown in this table.

■ **node_connection_accept= true|false**

Specify whether a connection is to be established from the extraction node master process to the extraction master process. The extraction node master process establishes connection with the extraction master process at the following times:

- When the extraction node master process is started by the `hdestart_n` command
- When the extraction node master process retries connection establishment after it was disconnected from the extraction master process

Specification of the `node_connection_accept` operand enables you to control the operation of the extraction master process and extraction node master process at the times noted above.

The following describes how the operation of the extraction master process and extraction node master process is controlled by the `node_connection_accept` operand:

Event	When true is specified	When false is specified
Extraction node master process was started by the <code>hdestart_n</code> command.	<ul style="list-style-type: none"> • Operation of the extraction master process Accepts a connection requests from the extraction node master process. • Operation of the extraction node master process Connects to the extraction master process, and then starts replication. 	<ul style="list-style-type: none"> • Operation of the extraction master process Does not accept a connection request from the extraction node master process. • Operation of the extraction node master process Terminates the process due to a timeout when the connection wait time specified in the <code>hdestart_n</code> command expires.
Extraction node master process was disconnected from the extraction master process.	<ul style="list-style-type: none"> • Operation of the extraction master process Waits for a connection request from an extraction node master process if the extraction master process is disconnected from all extraction node master processes. • Operation of the extraction node master process Reconnects the extraction master process. 	<ul style="list-style-type: none"> • Operation of the extraction master process Terminates the process if the extraction master process is disconnected from all extraction node master processes. • Operation of the extraction node master process Terminates the process after detecting the disconnection.

The `node_connection_accept` operand value is effective only when `server` is specified in the `nodecontrol` operand. If the HiRDB subject to extraction processing uses a standby-less system switchover (effects distributed) configuration, you must specify `true` in this operand.

When you specify `true` in the `node_connection_accept` operand, you must specify the `connection_accept_hostname` and `connection_accept_service` operands.

■ **`connection_accept_hostname=extraction-node-master-process-connection-request-accepting-host-name`**

~ <identifier of 1-32 characters>

Specify the host name used to accept a connection request from the extraction node master process when connection is to be established from extraction node master process to extraction master process.

If you have specified `true` in the `node_connection_accept` operand, you must specify this operand.

In the case of a standby-less system switchover (effects distributed) configuration, you must specify in this operand a host name that inherits IP addresses, because the connection request issued by the extraction node master

process to the extraction master process that is started in the standby system after system switchover uses the same host name as the primary system.

If the extraction node master process is to be started by the `hdestart_n` command, the host name specified in this operand must also be specified in the `-x` option as the name of the host that runs the extraction master process.

Note that the host name specified in this operand must be added to the OS's `hosts` file.

- **connection_accept_service=extraction-node-master-process-connection-request-accepting-service-name**

~ <identifier of 1-64 characters>

Specify the service name used to accept a connection request from the extraction node master process when connection is to be established from extraction node master process to extraction master process.

If you specified `true` in the `node_connection_accept` operand, you must specify this operand.

This service name (and port number) must be different from the service name (and port number) specified in the `mstservice` operand of the extraction system definition and the `hdeservice` operand of the transmission environment definition.

If the extraction node master process is to be started by the `hdestart_n` command, the service name specified in this operand must also be specified in the `-n` option.

Note that the service name specified in this operand must be added to the OS's `services` file.

- **connection_accept_waittime=extraction-node-master-process-connection-request-wait-time**

~ ((1-3600)) <<300>> (seconds)

Specify the amount of time to wait for the extraction master process to accept a reconnection request from an extraction node master process when the extraction master process is disconnected from all extraction node master processes.

If the time specified in this operand elapses before a connection request is received from any of the extraction node master processes, the extraction master process is terminated.

This operand is applicable only when `true` is specified in the `node_connection_accept` operand.

- **connection_retry_time=extraction-node-master-process-reconnection-processing-time**

~((1-3600)) <<300>> (seconds)

Specify the amount of time during which the extraction node master process is to attempt to reestablish connection with the extraction master process when the extraction node master process is disconnected from the extraction master process.

If the time specified in this operand elapses before the extraction node master process can reconnect to the extraction master process, the extraction node master process is terminated.

This operand is applicable only when `true` is specified in the `node_connection_accept` operand.

■ **node_syslogout= true|false**

Specify whether the information that is output to the extraction node master error information file is to be output also to the corresponding machine's syslog file.

`true`

Output to the syslog file the information that is output to the extraction node master error information file.

`false`

Do not output to the syslog file the information that is output to the extraction node master error information file.

The Windows edition does not support this operand because whether error information for each node is to be output is determined by the `syslogout` operand; if this operand is specified in the Windows edition, it is ignored.

Note:

When `true` is specified in this operand, the system outputs all error information for each node. Specifying `true` is not recommended, because a large amount of information might be output (depending on the operating environment), thereby consuming a large amount of syslog file resources.

■ **send_counter_reset = true|false**

Specify whether the data transmission count is to be reset when the source system starts.

`true`

Reset the data transmission count when the source system starts.

`false`

Do not reset data transmission count when the source system starts.

- **hirdb_audit_trail = all|none**

Specify whether an audit trail is to be collected if an operation subject to audit trail collection (audit event) occurs.

An audit event is generated when the `hdeevent` or `hdeprep` command or the extraction master process is executed.

`all`

Collect an audit trail.

`none`

Do not collect an audit trail.

If you want to suppress the overhead of having source Datareplicator processing collect an audit trail, we recommend that you specify `none`.

- **resource_chk_err = continue|stop**

Specify the action to be taken if an error is detected when the source Datareplicator checks file integrity and file sizes during initialization or start processing.

If you always perform initialization or start processing when no change has been made to the data linkage environment or operands, we recommend that you specify `stop`.

`continue`

Continue with the initialization or start processing if it's possible for data linkage to be resumed or if the user changed operands intentionally.

`stop`

Stop the initialization or start processing regardless of the nature of the error.

- **recover_info_send = true|false**

Specify whether recovery information is to be sent to the target Datareplicator.

`true`

Send recovery information.

`false`

Do not send recovery information.

Note:

We recommend that you specify `true` because if recovery information is not sent, data linkage recovery cannot be performed via the system log file. For details about data linkage recovery via the system log file, see *9.5 Data*

linkage recovery via the system log file.

■ **recover_info_send_interval = recovery-information-transmission-interval**

~((1-32767))<<1>>

Specify the interval at which recovery information is to be sent to the target Datareplicator, expressed as a number of transactions in the source system. This operand takes effect only when `true` is specified in the `recover_info_send` operand. However, during event transmission, recovery information is sent to the target Datareplicator regardless of this operand's setting.

For guidelines for the operand value, see 9.5.2(3)(a) *Creating a recovery information file*.

■ **cm_errno_check = true/false**

Specify whether the nature of a communication error is to be checked, and then connection establishment is to be retried only if reconnection is possible, or if connection establishment is to be retried unconditionally in the event of a communication error.

`true`

Retry connection establishment only if reconnection is possible.

Note that depending on the combination of network configuration and devices, connection establishment might not be retried even in response to a transient error. If the error prohibits reconnection, the transmission process terminates with an error.

`false`

Retry connection unconditionally, regardless of the nature of the error.

Notes:

- If `true` is specified, connection establishment is retried even for an error that is checked by the source Datareplicator. Therefore, a communication error that can be detected at an early stage when `true` is specified is not detected until connection establishment is retried as many times as specified in the `retrynum` operand in the transmission environment definition. To detect communication errors at an early stage, we recommend that you do not set the connection retry count (`retrynum` operand in the transmission environment definition) to unlimited.
- If connection establishment is retried in the event of an error that cannot result in reconnection, the `KFRB02053-W` message is output (the reason for line connection is `other`). Also, the system call type and `errno` are output to the extraction node master error information file as additional information.

■ **node_pddir=source-HiRDB's-PDDIR-environment-variable-value**

~ <pathname of 1 to 255 bytes>

If the source HiRDB is a parallel server and different information is specified in the `PDDIR` environment variable for each server machine, specify the absolute pathname of the source HiRDB's directory that is specified in the `PDDIR` environment variable. Specify this operand only in an individual definition section. If you omit the `node_pddir` operand, the source Datareplicator at the corresponding server machine will use the value of the `PDDIR` environment variable that is specified for the user environment in which the source Datareplicator's commands are executed.

■ **node_pdconfpath=HiRDB's-PDCONFPATH-environment-variable-value**

~ <pathname of 1 to 255 bytes>

If the source HiRDB is a parallel server and different information is specified in the `PDCONFPATH` environment variable for each server machine, specify the absolute pathname of the directory used to store the source HiRDB's system definition file that is specified in the `PDCONFPATH` environment variable. Specify this operand only in an individual definition section. If you omit the `node_pdconfpath` operand, the source Datareplicator at the corresponding server machine will use the value of the `PDCONFPATH` environment variable that is specified for the user environment in which the source Datareplicator's commands are executed.

■ **node_shlibpath=HiRDB's-SHLIB_PATH-environment-variable-value**

~ <pathname of 1 to 255 bytes>

If the source HiRDB is a parallel server and different information is specified in the `SHLIB_PATH` environment variable for each server machine, specify the absolute pathname of `$PDDIR/lib` that is specified in the `SHLIB_PATH` environment variable. Specify this operand only in an individual definition section. If you omit the `node_shlibpath` operand, the source Datareplicator at the corresponding server machine will use the value of the `SHLIB_PATH` environment variable that is specified for the user environment in which the source Datareplicator's commands are executed.

■ **node_host=node-master-host-name**

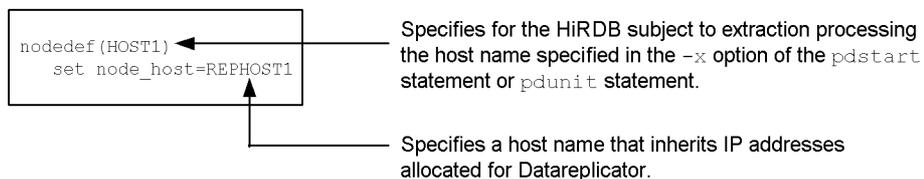
~ <1-32 alphanumeric characters>

If a system switchover configuration is used and the conditions listed below are not satisfied, you must allocate for Datareplicator a host name that inherits IP addresses and specify that host name in this operand:

- In the system common definition for the HiRDB subject to extraction processing, a host name that inherits IP addresses is specified in the `-x` option of the `pdunit` or `pdstart` statement.

- The system is set up in such a manner that, pursuant to the HA monitor's resource server facility, the above IP address is switched over after the HiRDB subject to extraction processing and the source Datareplicator have been terminated.

The following is an example of specifying this operand:



5.3 Extraction environment definition

An extraction environment definition specifies information needed to execute extraction processing. If the source HiRDB is a parallel server, you can define extraction environments for different back-end servers using one extraction environment definition file.

5.3.1 Format

```
[ set dsid=data-linkage-identifier ]
[ set qufile001="extraction-information-queue-filename" ]
[ set qufile002="extraction-information-queue-filename"
  [... [ set qufile016="extraction-information-queue-filename" ]]]
[ set queuesize=extraction-information-queue-file-size ]
[ set logiosize=system-log-I/O-buffer-size ]
[ set quiosize=extraction-information-queue-I/O-buffer-size ]
[ set extsuppress=true|false ]
[ set
ext_wait_interval=extraction-restart-interval-after-detecting-end-of-extraction ]
[ set extact_level=current_gen|original_gen|all_gen ]
[ set int_trc_getv=true|false ]
[ set qufullwarn=extraction-information-queue-file-full-warning-value ]

[ set extract_delay_limit_time=extraction-delay-period-threshold]]
```

Items defined in individual definition sections only

```
[ set device01=Datareplicator-file-system-area-name [ [ ,allocation-file-type ] ... ]
[ set device02=Datareplicator-file-system-area-name [ [ ,allocation-file-type ] ... ]
  [... [ set
device18=Datareplicator-file-system-area-name [ [ ,allocation-file-type ] ... ]]]]]
```

5.3.2 Modifying defined information

If the source Datareplicator has terminated abnormally, start it normally, and then perform the procedure to modify the defined information.

To modify defined information:

1. Terminate the source Datareplicator.
2. Use a text editor to modify the defined information.
3. Start the system in a start mode that will apply the modified information, as shown in the following table.

Table 5-4: Start mode that applies the modified extraction environment definition

Operand name	Start mode that applies the modified information	
	Initial start ^{#1}	Normal start ^{#1}
dsid	Y	--
qufilexx ^{#2}	Y	--
queuesize	Y	--
logiosize	Y	Y
quiosize	Y	--
extsuppress	Y	--
ext_wait_interval	Y	Y
extract_level	Y	Y
int_trc_getv	Y	Y
qufullwarn	Y	Y
extract_delay_limit_time	Y	Y
devicexx ^{#3}	Y	--

Legend:

Y: The start mode denoted by this column applies the operand's modified information. If the start modes of both columns apply a particular operand's modified information, either of those start modes can be used to apply the modified information.

--: Not applicable

#1

This is the start mode used to start the source Datareplicator.

#2

xxx is a value in the range from 001 to 016.

#3

xx is a value in the range from 01 to 18.

5.3.3 Explanation of the operands

- **dsid=data-linkage-identifier**

~ <hexadecimal> ((00-FF)) <<00>>

Specify the data linkage identifier that will enable the target system to identify the source of the update information. This data linkage identifier must be unique in the corresponding target system.

- **qfile001="extraction-information-queue-filename"**

~ <[*pathname*/]*filename* of 1-64 bytes> <<\${HDEPATH}/qfile001>>

- **qfile002="extraction-information-queue-filename"**

~ <[*pathname*/]*filename* of 1-64 bytes> <<\${HDEPATH}/qfile002>>

:

- **qfile016="extraction-information-queue-filename"**

~ <[*pathname*/]*filename* of 1-64 bytes> <<\${HDEPATH}/qfile016>>

Specify the names of the extraction information queue files, as absolute or relative pathnames. If you specify a relative pathname, Datareplicator assumes `${HDEPATH}/relative-pathname` as the absolute pathname. The source Datareplicator adds `_server-name` to the specified name and creates a file under this filename. For UNIX, if the file is a character special file, you must create a file with the specified name to which `_server-name` is appended.

If you specify an absolute pathname, the complete filename including `_server-name` must not exceed 64 bytes. If you specify a relative pathname, the assumed absolute pathname plus `_server-name` must not exceed 125 bytes. This filename must be unique in the source system. If the source HiRDB is a parallel server, the filename must be unique within the back-end server.

Datareplicator uses file swapping to store information in the extraction information queue files. Therefore, you must provide at least two extraction information queue files; the maximum is 16 files. If you do not specify any extraction information queue filenames, Datareplicator assumes `${HDEPATH}/qfile001` for `qfile001` and `${HDEPATH}/qfile002` for `qfile002`.

You must specify the `qfile001` to `qfile016` operands consecutively in ascending order beginning with `qfile001`. If they are not consecutive or in ascending order, Datareplicator stores update information in only those extraction information queue files that are specified consecutively in ascending order from the beginning. Datareplicator swaps the files in the order of their specification in this operand.

■ **queuesize=extraction-information-queue-file-size**

~ <unsigned integer> ((33-1000000000)) <<65>> (KB)

Specify the file size of the extraction information queue files specified with the `qufile001` to `qufile016` operands. The size specified in the `queuesize` operand applies to each of the files specified with the `qufile001` to `qufile016` operands.

For details about the formula for determining the size of an extraction information queue file, see *4.6.8 Designing the source Datareplicator's resources*.

If you handle the extraction information queue files as large files, specify at least 2097152 (2 GB). To handle large files, you must already have set the OS and Datareplicator file system areas to support large files. For details, see *6.11 Handling of large files*.

Note:

If you change the size of an existing extraction information queue file, you must initialize the source Datareplicator. Therefore, before you change file sizes, make sure that replication has been completed.

■ **logiosize=system-log-I/O-buffer-size**

~ <unsigned integer> ((32-510)) <<510>> (KB)

Specify the size of the system log I/O buffer that is used to read update information from the source HiRDB's system log. The actual buffer size is determined by the following formula using this operand:

$$4 \times \uparrow \text{logiosize} / 4 \uparrow$$

For details about the size of the system log I/O buffer, see *4.6.3 Designing the extraction procedure*.

■ **quiosize=extraction-information-queue-I/O-buffer-size**

~ <unsigned integer> ((32-510)) <<32>> (KB)

Specify the size of the extraction information queue I/O buffer that is used to store update information in the extraction information queue file or to read update information from the extraction information queue file. For details about the size of the update information queue I/O buffer, see *4.6.3 Designing the extraction procedure*.

■ **extsuppress=true|false**

Specify whether to use of the source Datareplicator for a server that contains no tables subject to extraction processing is to be suppressed. The `extsuppress` operand is applicable only when the source HiRDB is a parallel server. Specify `true` to reduce the workload at a server that contains no tables subject to

extraction processing. To ignore whether each server contains tables subject to extraction processing, specify `false`. For details about the specification of the `extsuppress` operand and the source HiRDB and source Datareplicator processing, see *6.5.4 Processing at the source HiRDB and source Datareplicator depending on the specification of `extsuppress` in the extraction environment definition*.

`true`

The source Datareplicator is not to operate at a server that contains no tables subject to extraction processing. This means that no update information will be extracted from such a server even if a table it stores is specified in an extraction definition.

`false`

The source Datareplicator is to operate at all servers, including servers that contain no tables subject to extraction processing. If any server contains a table that is specified in an extraction definition, Datareplicator will extract update information from that table. If no table stored at a particular server is specified in an extraction definition, Datareplicator will not extract update information at that server. If you specify `false` and continue executing the HiRDB Datareplicator linkage facility with only the source HiRDB and without starting the source Datareplicator, the system log file might become full even though a corresponding server contains no tables subject to extraction processing.

■ **`ext_wait_interval=extraction-restart-interval-after-detecting-end-of-extraction`**

~ <unsigned integer> ((100-60000)) <<5000>> (milliseconds)

Specify the interval after which extraction processing is to be restarted once the end of update information sent from a DBMS subject to extraction processing is detected.

By specifying a small value in this operand, you can improve the spontaneity of extraction processing, because there is only a brief wait after the end of the system log file is detected. However, this might have an adverse effect on performance, because the CPU utilization factor increases. You need to take into account the frequency of update processing and CPU performance when specifying this value.

■ **`extract_level= current_gen|original_gen|all_gen`**

Specify the extraction level when the source HiRDB uses the inner replica facility.

`current_gen`

If the source HiRDB uses the inner replica facility, extract only the updates to the current RDAREA.

`original_gen`

If the source HiRDB uses the inner replica facility, extract only the updates to the original RDAREA.

`all_gen`

If the source HiRDB uses the inner replica facility, also extract the updates to all replica RDAREAs (including the original RDAREA).

■ **`int_trc_getv=true|false`**

Specify whether activity trace information is to be collected for extraction-related processes at this node (`commondef` common definition section) or back-end server (`besdef` individual definition section).

`true`

Collect activity trace for each extraction-related process.

`false`

Do not collect activity trace for each extraction-related process.

This operand is applicable only when `nc` is specified in the second parameter of the `int_trc_lvl` operand in the extraction system definition.

■ **`qufullwarn=extraction-information-queue-file-full-warning-value`**

Specify this operand to issue a warning message before the extraction information queue file becomes full. When the number of available extraction information queue files becomes equal to or less than the value specified here, Datareplicator issues a warning message. When this value is equal to or greater than the number of extraction information queue files, Datareplicator issues a warning message each time the extraction information queue files swap.

■ **`extract_delay_limit_time=extraction-delay-period-threshold`**

~ <unsigned integer> ((0-86400)) <<0>> (seconds)

Specify the threshold value for the difference (extraction delay period) between the time at which update information is stored in the system log file and the time at which the extraction information is written into the extraction information queue file. Whenever the actual extraction delay period exceeds the threshold value specified here, a warning message is output.

You must take into account the delay period also when you specify the `ext_wait_interval` operand in the extraction environment definition. If you specify a value of 0, the delay monitoring facility is disabled for extraction processes.

Note:

Adjust the value of this operand, taking into account the amount of update information subject to replication, the transmission interval, and the import interval. A recommended guideline value to start with is 10 minutes (600 seconds).

■ **device01=Datareplicator-file-system-area-name[[,allocation-file-type] ...]**

~ <absolute pathname of 1-125 bytes>

device02=Datareplicator-file-system-area-name[[,allocation-file-type] ...]

~ <absolute pathname of 1-125 bytes>

:

device18=Datareplicator-file-system-area-name[[,allocation-file-type] ...]

~ <absolute pathname of 1-125 bytes>

Specify the names of the Datareplicator file system areas and the file types to be allocated in each file system area. Specify a two-digit sequence number for *xx* in *devicexx*. The following table shows the correspondence between files and the file types to which they are allocated:

Allocation file type	File name to be specified in each operand
qufilexxx (xxx: integer from 001 to 016)	Link file name for extraction information queue file that was specified in set qufilexx. The server name is assigned to a created link file.
hde_file	Link file name for the corresponding back-end server's data linkage file.
sts_file	Link file name for the corresponding back-end server's status file.

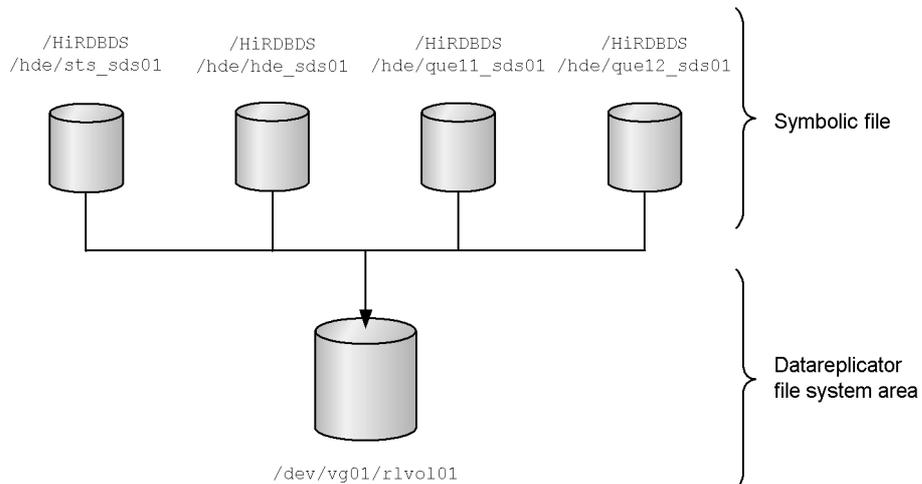
If you do not specify any allocation file types, all files at the corresponding back-end server become subject to allocation. When a file system area is specified in a *devicexx* operand, Datareplicator automatically creates it and links it to the Datareplicator file system. You must ensure that the name of the file corresponding to an allocation file type has not already been allocated as a character special file.

You can specify the *devicexx* operand only in an individual definition section. If you specify it in the common definition section, Datareplicator detects an error during analysis of extraction definitions and cancels the subsequent processing. If the source HiRDB employs the single server configuration, specify the single server name as the server name at the start of the individual definition section (*besdef*).

The following is a definition example of the `devicexx` operand:

- Definition example 1: For a single server

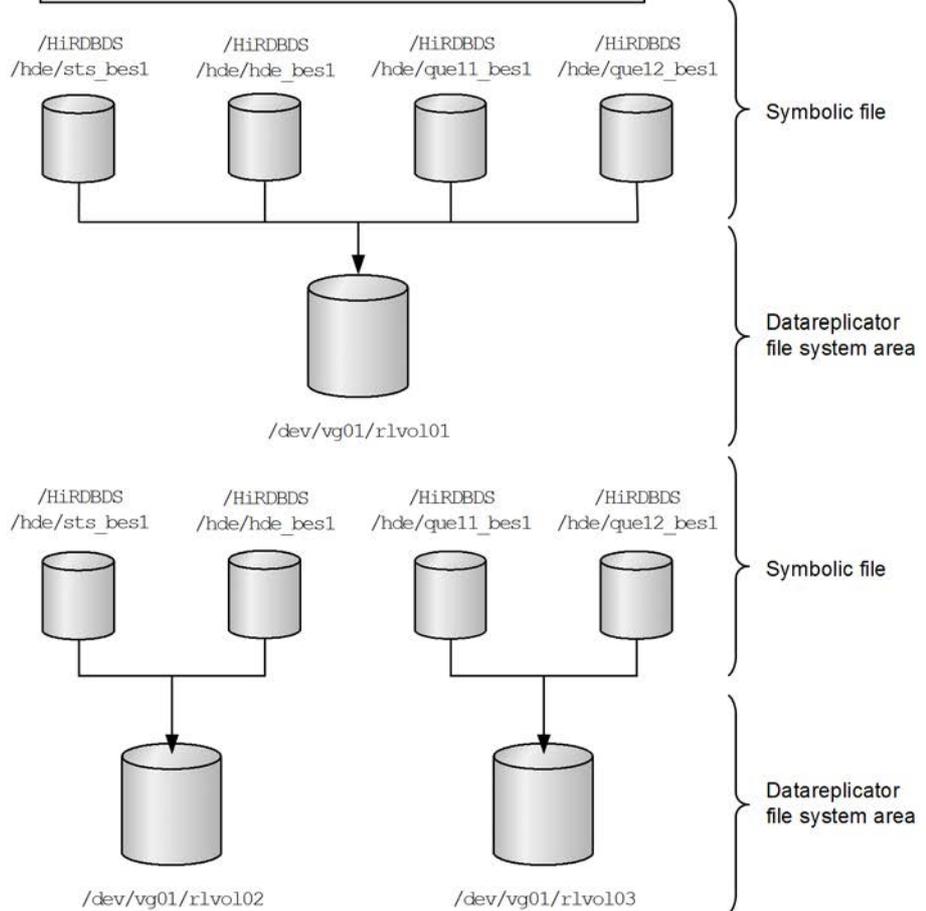
```
set qufile001 = /HiRDBDS/hde/que11
set qufile002 = /HiRDBDS/hde/que12
set device01  = /dev/vg01/rlvol01
```



• Definition example 2: For a parallel server

```

besdef (bes1)
set qufile001 = /HiRDBDS/hde/que11
set qufile002 = /HiRDBDS/hde/que12
set device01 = /dev/vg01/rlvol01
besdef (bes2)
set qufile001 = /HiRDBDS/hde/que11
set qufile002 = /HiRDBDS/hde/que12
set device01 = /dev/vg02/rlvol02,hde_file,sts_file
set device02 = /dev/vg03/rlvol03,qufile001,qufile002
    
```



Note:

To omit a `devicexx` operand from the definition after Datareplicator has been operated using the `devicexx` operand:

1. Stop Datareplicator.

2. Delete the `devicexx` operand from the extraction environment definition.
3. Use the `hdsfmkfs` command to initialize the Datareplicator file system area that was specified with the deleted `devicexx` operand.
4. Execute initial start on Datareplicator.

If you delete a `devicexx` operand from the definition, and then execute initial start of Datareplicator without following the above procedure, an invalid file error might occur during execution.

5.4 Transmission environment definition

A transmission environment definition specifies information needed to execute transmission processing. You must create a transmission environment definition file for each target identifier. If the source HiRDB is a parallel server, you can define transmission environments for different back-end servers using one transmission environment definition file.

5.4.1 Format

```
[ set sendhdsid=target-system-identifier ]
[ set hdeservice=service-name ]
[ set hdehost=target-host-name ]
[ set protocol=tcp|osi ]#
[ set extract_tselector=T-selector ]#
[ set nsap_address=NSAP-address ]#
[ set senduoc=use|nouse ]
[ set sendintvl=transmission-interval ]
[ set sendintvl_scale=minute|second ]
[ set nsndid001=transmission-suppressed-original-receiver-identifier
  [... [ set nsndid256=transmission-suppressed-original-receiver-identifier ]]]
[ set keepalive=true|false ]
[ set retrynum=connection-retries-count ]
[ set retry_interval=connection-retry-interval ]
[ set overwrite=true|false ]
[ set overwrite_continue=true|false ]
[ set maxtran=maximum-concurrently-executable-transactions-count ]
[ set maxtrandata=maximum-update-information-items-per-transaction ]
[ set readbufnum=extraction-information-queue-I/O-buffers-count-for-
transmission ]
[ set editbufsize=update-information-editing-buffer-size ]
[ set prg_eventno=event-number ]
[ set int_trc_getv=true|false ]
[ set queue_read_wait_interval=transmission-process's-extraction-information-
queue-file-read-interval]
[ set recvwatchtime=data-reception-line-monitoring-interval]
[ set send_delay_limit_time=transmission-delay-period-threshold]
[ set reflect_mode=server|uap ]
[ set eventsync=synchronous-event-code ]
[ set eventcntreset=data-transmission-count-reset-event-code ]
```

#: Specify this operand if you use the OSI protocol for communication. This operand is applicable only to the HP-UX edition of Datareplicator (excluding the IPF version).

5.4.2 Modifying defined information

To modify defined information:

1. Terminate the source Datareplicator.

To execute an initial start subsequently, terminate the source Datareplicator normally. If the source Datareplicator has terminated abnormally, start it normally, and then perform the procedure to modify the defined information.

2. Use a text editor to modify the defined information.
3. Start the system in a start mode that will apply the modified information, as shown in the following table.

Table 5-5: Start mode that applies the modified transmission environment definition

Operand name	Start mode that applies the modified information	
	Initial start or partial initial start ^{#1}	Normal start ^{#1}
sendhdsid	Y	--
hdeservice	Y	--
hdehost	Y	Y
protocol	Y	--
extract_tselector	Y	--
nsap_address	Y	--
senduoc	Y	Y
sendintvl	Y	Y
sendintvl_scale	Y	Y
nsndidxxx ^{#2}	Y	--
keepalive	Y	Y
retrynum	Y	Y
retry_interval	Y	Y
overwrite	Y	Y
overwrite_continue	Y	Y
maxtran	Y	Y

Operand name	Start mode that applies the modified information	
	Initial start or partial initial start ^{#1}	Normal start ^{#1}
maxtrandata	Y	Y
readbufnum	Y	Y
editbufsize	Y	Y
prg_eventno	Y	Y
int_trc_getv	Y	Y
queue_read_wait_interval	Y	Y
recvwatchtime	Y	Y
send_delay_limit_time	Y	Y
reflect_mode	Y	--
eventsync	Y	--
eventcntreset	Y	Y

Legend:

Y: The start mode denoted by this column applies the operand's modified information. If the start modes of both columns apply to a particular operand's modified information, any of the start modes can be used to apply the modified information.

--: Not applicable

#1

This is the start mode used to start the source Datareplicator.

#2

xxx is a value in the range from 001 to 256.

5.4.3 Explanation of the operands

- **sendhdsid=target-system-identifier**

~ <hexadecimal> ((00-FF)) <<00>>

Specify the identifier of the data linkage target system. If the target system is the target Datareplicator, specify the target Datareplicator identifier. If the target system is XDM/DS, specify the target XDM/DS identifier in EBCDIK converted

to hexadecimal.

■ **hdeservice=service-name**

~ <identifier of 1-64 characters> <<hirdbds>>

Specify the service name that was specified when the source Datareplicator's communication entry was added to the `services` file.

Example

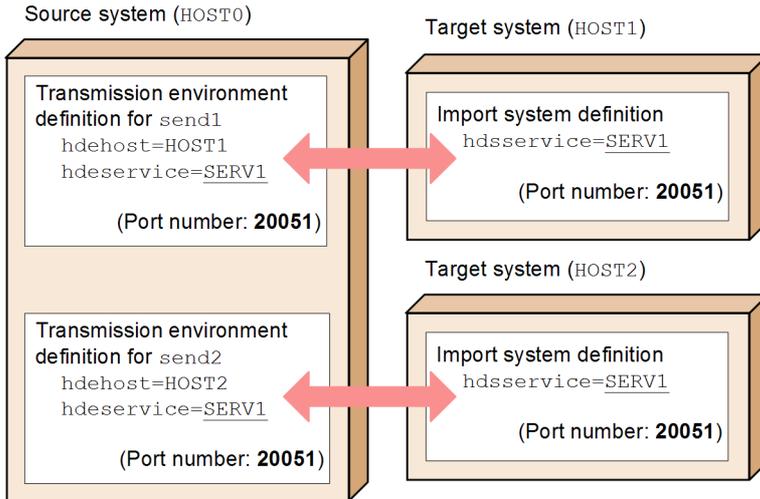
This example target system consists of two server machines.

Provide a transmission environment definition for each server machine. For the service name, specify the service name that was specified when the source system's communication environment was configured. Make sure that the port number is the same as that of the target server machine.

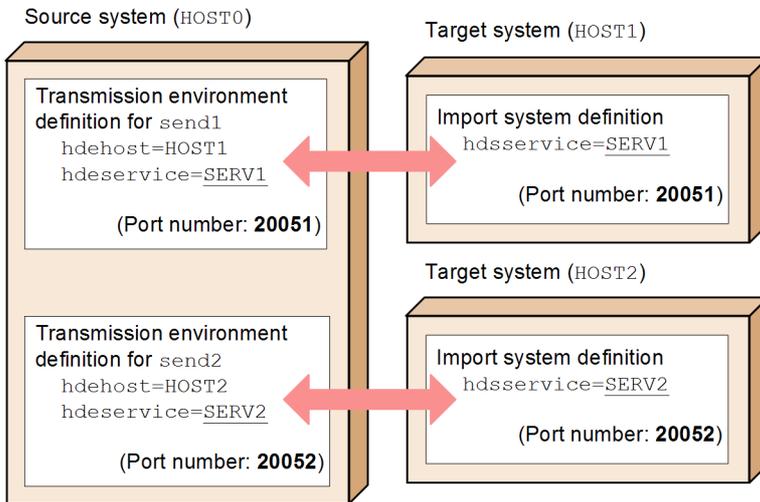
The service name can be the same even if the target system's server machines have different target host names.

The following figure shows an example specification of a transmission environment definition when the target system consists of two server machines.

- When the port number is the same as that of the target server machine



- When the port number is different from that of the target server machine



■ **`hdehost=target-host-name`**

~ <identifier of 1-255 characters> <<hdehost>>

Specify the host name that was specified when the target entry was added to the `hosts` file or DNS. Specify the host name as a string containing only alphanumeric characters, the dash (-), underline (_), and period (.) and beginning

with an alphabetic character.

■ **protocol=tcp|osi**

Specify the protocol to be used for communications with the target system.

`tcp`

Use the TCP/IP protocol for communications.

`osi`

Use the OSI protocol for communications. If you specified `sendmst` in the `sendcontrol` operand in the extraction system definition, you cannot use the OSI communications protocol.

This operand is applicable only to the HP-UX edition of Datareplicator (excluding the IPF version).

■ **extract_tselector=T-selector**

~ <even number of hexadecimal digits with a length of 2-64 digits> <<00>>

Specify the same T-selector as for the target Datareplicator's reception process at the destination. Specify this operand if you use the OSI protocol for communication. Note that this operand is applicable only to the HP-UX edition of Datareplicator (excluding the IPF version).

■ **nsap_address=NSAP-address**

~ <even number of hexadecimal digits with a length of 2-40 digits> <<00>>

Specify the NSAP address of the machine where the target Datareplicator is running at the destination. Specify this operand if you use the OSI protocol for communication. Note that this operand is applicable only to the HP-UX edition of Datareplicator (excluding the IPF version).

■ **senduoc=use|nouse**

Specify whether a transmission data UOC routine is to be used.

`use`

Use a transmission data UOC routine. If in the case of UNIX Datareplicator, there is no `libsenduoc.sl` under `SHLIB_PATH`, Datareplicator assumes `nouse`. If, in the case of Windows Datareplicator, there is no `senduoc.dll` under `PATH`, a DLL loading error occurs and transmission stops.

`nouse`

Do not use a transmission data UOC routine.

■ **sendintvl=transmission-interval**

~ <unsigned integer> ((0-1440)) <<5>>

Specify the transmission interval to be used for sending extracted update information to the target system. You use the `sendintvl_scale` operand to specify the units of the transmission interval. If you specify 0, Datareplicator sends the update information in units of transactions. If you want to reduce the time leading up to import processing, start tuning from one second. Specifying 0 might slow down the transmission processing.

When applying data linkage to tables for which the `WITHOUT ROLLBACK` option is specified:

For tables for which the `WITHOUT ROLLBACK` option is specified, one update operation is treated as one transaction. If 0 is specified in this operand, transmission processing occurs each time a table for which the `WITHOUT ROLLBACK` option is specified is updated, resulting in a large amount of transmission processing overhead. If you apply data linkage to tables for which the `WITHOUT ROLLBACK` option is specified, we recommend that you specify a nonzero value in this operand.

■ **`sendintvl_scale=minute|second`**

Specify the units of the `sendintvl` operand value.

`minute`

Assume that the value of the `sendintvl` operand is in minutes.

`second`

Assume that the value of the `sendintvl` operand is in seconds.

This operand is applicable only when the `sendintvl` operand is specified. To specify this operand in an individual definition section, you must specify the `sendintvl` operand again in the same `besdef`. If you omit the `sendintvl` operand, Datareplicator ignores the `sendintvl_scale` operand.

The following table shows the combinations of `sendintvl` and `sendintvl_scale` operands and their relationship with a valid value:

Specification in an individual definition section		Specification in the common definition section		
		sendintvl specified		sendintvl omitted (sendintvl_scale is ignored)
		sendintvl_scale specified	sendintvl_scale omitted	
One sendintvl specified	sendintvl_scale specified	Interval: Value specified in the individual definition section is effective. Units: Value specified in the individual definition section is effective.	Interval: Value specified in the individual definition section is effective. Units: Value specified in the individual definition section is effective.	Interval: Value specified in the individual definition section is effective. Units: Value specified in the individual definition section is effective.
	sendintvl_scale omitted	Interval: Value specified in the individual definition section is effective. Units: Minutes is assumed.	Interval: Value specified in the individual definition section is effective. Units: Minutes is assumed.	Interval: Value specified in the individual definition section is effective. Units: Minutes is assumed.
sendintvl omitted (sendintvl_scale is ignored)		Interval: Value specified in the common definition section is effective. Unit: Value specified in the common definition section is effective.	Interval: Value specified in the common definition section is effective. Unit: Minutes is assumed.	Interval: 5 is assumed. Unit: Minutes is assumed.

■ **nsndid001=transmission-suppressed-original-receiver-identifier [... [nsndid256=transmission-suppressed-original-receiver-identifier]]**

~ <hexadecimal> ((00-FF))

Specify the identifiers of original receivers that are to be subject to suppression of update information transmission. You use this operand to prevent loopback in a data linkage system that consists of multiple systems, each of which has both a source and a target of transmission.

Specify in each *transmission-suppressed-original-receiver-identifier* the source Datareplicator identifier at a system that originally receives update information that was received by a target Datareplicator running under this source HiRDB (the source system that sent update information to this target system). For details about

suppression of loopback, see *4.5 Designing the data linkage system mode*.

You must specify the `nsndid001` to `nsndid256` operands consecutively in ascending order beginning with `nsndid001`. If they are not consecutive or in ascending order, Datareplicator uses only those identifiers that are specified consecutively in ascending order from the beginning. If the `nsndid` operand is omitted, Datareplicator sends all intended update information to the target system specified in the `sendhdsid` operand.

■ **keepalive=true|false**

Specify whether the `keepalive` option can be specified for the socket.

`true`

Specify the `keepalive` option.

`false`

Do not specify the `keepalive` option.

■ **retrynum=connection-retries-count**

~ <unsigned integer> ((0-256)) <<0>>

Specify the number of times communication establishment with the target system can be retried for transmission processing if it fails. Datareplicator attempts to reestablish communication with the target system up to the specified number of times. If 0 is specified, Datareplicator retries until connection is established successfully or a stop request is accepted.

The reconnection interval depends on the `sendcontrol` operand value. The following table shows the reconnection interval:

sendcontrol operand value	Reconnection interval
<code>nodemst</code>	Connection retry interval (<code>retry_interval</code> operand value)
<code>sendmst</code>	Transmission interval (<code>smt_sendintvl</code> operand value)

■ **retry_interval=connection-retry-interval**

~ <unsigned integer>((1-60))<<60>>(seconds)

Specify a connection retry interval if you want connection to be reestablished immediately in the event of a system switchover in the target system. This operand is applicable only when `nodemst` is specified in the `sendcontrol` operand.

The interval specified in this operand is also applied to retries in the event connection is lost because the import information queue file has become full. Note that when the import information queue file becomes full, a message

reporting that the queue file is full is output at this retry interval, so the message might be output frequently if the specified value is too small.

■ **`overwrite=true|false`**

Specify whether transmission processing to the target system specified in `sendhdsid` is to be placed in the reduced mode when the extraction information file queue becomes full.

If you specified `uap` in the `reflect_mode` operand in the transmission environment definition, specify `false`. If you specify any other value in this operand in such a case, the `KFRB00847-E` message will be output.

`true`

Apply reduced-mode transmission processing to the target system. When the extraction information file queue becomes full, Datareplicator cancels the corresponding transmission processing and discards the update information subject to transmission from the extraction information queue file. In this case, the target system at the corresponding destination requires re-creation of the target database because conformity between the source and target databases is lost.

`false`

Do not apply the reduced-mode transmission processing to the target system.

■ **`overwrite_continue=true|false`**

Specify whether the reduced mode specified with the `overwrite` operand is to be inherited the next time transmission processing is restarted.

`true`

When transmission processing is restarted in the reduced mode, inherit the reduced mode by detecting a transmission process start error.

`false`

When transmission processing is restarted in the reduced mode, release the reduced mode and restart transmission processing from the transaction that was extracted after the restart.

■ **`maxtran=maximum-concurrently-executable-transactions-count`**

`~ <unsigned integer> ((5-1780000)) <<100>>`

Specify the number of transactions that can be executed concurrently to determine the initial value of the transaction management information buffer that will be used to edit and send update information. The initial value of the transaction management information buffer is obtained internally as the product of the maximum number of concurrently executable transactions specified with the

`maxtran` operand and the maximum number of update information items per transaction specified with the `maxtrandata` operand. Specify the `maxtran` and `maxtrandata` operands so that their product does not exceed 89000000. For details about the size of the transaction management information buffer, see 4.6.4 *Designing the transmission procedure*.

■ **maxtrandata=maximum-update-information-items-per-transaction**

~ <unsigned integer> ((5-17800000)) <<500>>

Specify the maximum number of update information items that can be handled in a single transaction to determine the initial value of the transaction management information buffer that will be used to edit and send update information. If you have specified the `ukey` clause in the `extract` statement in the extraction definition, double the value to take into account the number of update information items for which pre-update data is stored.

The initial value of the transaction management information buffer is obtained internally as the product of the maximum number of concurrently executable transactions specified with the `maxtran` operand and the maximum number of update information items per transaction specified with the `maxtrandata` operand. Specify the `maxtran` and `maxtrandata` operands so that their product does not exceed 89000000. For details about the size of the transaction management information buffer, see 4.6.4 *Designing the transmission procedure*.

■ **readbufnum=extraction-information-queue-I/O-buffers-count-for-transmission**

~ <unsigned integer> ((1-255)) <<1>>

Specify the number of extraction information queue I/O buffers for transmission that are to be used to read update information from the extraction information queue file when update information is to be transmitted. Use the `quiosize` operand in the extraction environment definition to specify the size of the extraction information queue I/O buffers for transmission. For details about the extraction information queue I/O buffers for transmission, see 4.6.4 *Designing the transmission procedure*.

■ **editbufsize=update-information-editing-buffer-size**

~ <unsigned integer> ((1-2097151)) <<300>> (KB)

Specify the size of the update information editing buffer that is used to edit update information read from the extraction information queue file into the format supported by the target system. For details about the size of the update information editing buffer, see 4.6.4 *Designing the transmission procedure*.

You can reduce the number of communications by specifying a large value in this operand. We recommend that you specify a value that is greater than the amount of data (amount of update information) that is written to the extraction

information queue file within the amount of time specified in the `sendintvl` operand.

To determine the amount of data that is written to the extraction information queue file, execute periodically the `hdestate` command and measure the amount of increase in the offset information indicated by the Queue write position.

■ **prg_eventno=event-number**

~ <unsigned integer> ((0-255))

Specify the number of the event to be sent to the target system when the transmission process detects `PURGE TABLE` update information for a partitioned table spanning multiple back-end servers. The transmission process's action depends on the event number that you specify here. The following table shows the event numbers and the corresponding transmission process actions:

Specified event number	Transmission process's action
Not specified	Ignores the corresponding <code>PURGE TABLE</code> and continues transmission processing.
0	Sends a normal termination log to the target system, and then terminates transmission processing (transmission process stops).
Other	Sends the event corresponding to the number specified in the <code>prg_eventno</code> operand to the target system, and then resumes transmission processing.

■ **int_trc_getv=true|false**

Specify whether activity trace information is to be collected for transmission processes on this node (applicable to the `commondef` common definition section) or this back-end server (applicable to a `besdef` individual definition section).

`true`

Collect activity trace for each transmission process.

`false`

Do not collect activity trace for each transmission process.

This operand is applicable only when `nc` is specified in the second parameter of the `int_trc_lvl` operand in the extraction system definition.

■ **queue_read_wait_interval=transmission-process's-extraction-information-queue-file-read-interval**

~ <unsigned integer> ((100-60000)) <<2000>> (milliseconds)

Specify the interval after which the next read operation is to be restarted once the end of the extraction information queue file is detected under the control of the transmission process.

By specifying a small value in this operand, you can improve the spontaneity of transmission processing because of the brief wait time after the end of the extraction information queue file is detected. However, this might have an adverse effect on performance because the CPU utilization factor increases. You must take into account the frequency of update processing and CPU performance in specifying this value.

■ **recvwatchtime = data-reception-line-monitoring-interval**

~ <unsigned integer> ((1-35791394)) <<10>> (minutes)

Specify the line monitoring interval between extraction and import. If there is no response within the time specified in this operand, Datareplicator closes the line and retries connection establishment.

A smaller value detects line errors at an early stage. However, if the value is too small, unnecessary line disconnects and reconnects occur in cases such as the following. If this happens, you will need to adjust the operand value:

- The target system's workload is high and it takes time to store the transmitted data in the import information queue files.
- There is a high `editbufsize` operand value in the transmission environment definition and the amount of data to be written into the import information queue file at one time is large.
- The workload of the communication line is high and it takes time to respond from the target.

■ **send_delay_limit_time=transmission-delay-period-threshold**

~ <unsigned integer> ((0-86400)) <<0>> (seconds)

Specify the threshold value for the difference (transmission delay period) between the time at which update information is stored in the system log file and the time at which update data is sent and the target DBMS completes its reception. If the transmission delay period is greater than the threshold value specified here, a warning message is output.

You must also take into account the values of the `queue_read_wait_interval` and `sendintvl` operands in the transmission environment definition. If you specify a value of 0, the delay monitoring facility is disabled for transmission processes.

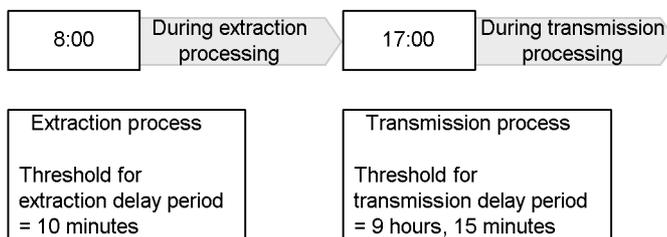
Note:

Adjust the value of this operand taking into account the amount of update information subject to replication, the transmission interval, and the import interval. A recommended guideline value to start with is 15 minutes (900 seconds).

Example

The following figure shows an example of operation when the start of transmission processing is delayed.

Figure 5-6: Example of operation when the start of termination processing is delayed



This example executes only extraction processing from 8:00 to 17:00 and starts transmission processing after 17:00. To monitor the transmission delay period in such an operating environment, you must take into account that update data is not transmitted for up to 9 hours, during which period it is accumulated at the target system. Therefore, a period of 9 hours must be added to the threshold for the transmission delay period.

- **reflect_mode= server|uap**

Specify whether the import transaction synchronization facility is to be used at the target Datareplicator.

`server`

Do not use the import transaction synchronization facility.

`uap`

Use the import transaction synchronization facility.

When you specify `uap` in this operand, you must specify the `eventsync` operand; you must also specify `false` in the `overwrite` operand.

- **eventsync=synchronous-event-code**

~ <unsigned integer> ((1-128))

Specify the event code for a synchronous event.

To specify this operand, you must specify `uap` in the `reflect_mode` operand.

Specify in this operand a value that is different from event codes specified in the `eventcntreset` and `prg_eventno` operands. An error will result if a duplicate value is specified.

- **eventcntreset=data-transmission-count-reset-event-code**

~ <unsigned integer> ((1-128))

Specify the event code for resetting the data-transmission count. To synchronize resetting at the source and target Datareplicators, this value must be the same as the `eventcntreset` operand value in the import environment definition.

5.5 Extraction definition

This section explains the definition of information required for the source Datareplicator's extraction and transmission processing.

5.5.1 Structure and format

(1) Structure

Figure 5-7 *Structure of the extraction definition* shows the structure of the extraction definition, and Table 5-6 *Contents of the extraction definition and the permitted numbers of specifications* shows the contents of the extraction definition and the permitted numbers of specifications.

Figure 5-7: Structure of the extraction definition

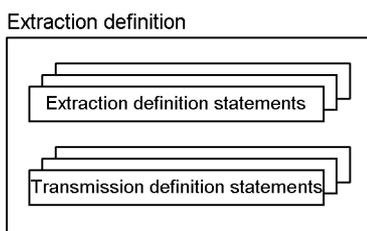


Table 5-6: Contents of the extraction definition and the permitted numbers of specifications

Definition name	Definition statement	Permitted number of specifications	Description
Extraction definition	extract statement	1-4096	Defines a table, columns, and mapping key subject to extraction processing.
Transmission definition	send statement	0-4096	Defines a destination of update information.

(2) Format

```

/* extraction definition statement */
{{ extract authorization-identifier.table-identifier( {column-name[ { , column-
name } } ... ] | * } )
  to update-information-name
{key|ukey}(column-name[ { , column-name } } ... ) [check
{not_null_unique|unique|none}] } } ...
/* transmission definition statement */
[ { send target-identifier from update-information-name
  [ where column-name {relational-operator literal|in(literal[ , literal] ... )
  
```

```

|flike(comparison-start-position, literal) }
[ and column-name {relational-operator literal|in(literal[ , literal] ...)
|flike(comparison-start-position, literal)}]]
}} ...]
;

```

5.5.2 Modifying defined information

If the source Datareplicator has terminated abnormally, start it normally, and then modify the defined information.

To modify defined information:

1. Terminate the source Datareplicator.
2. Use a text editor to modify the defined information.
3. At the source system, execute the `hdprep` command.
4. Start the source Datareplicator normally.

5.5.3 Extraction definition statement

An extraction definition statement defines extraction conditions for a table that is subject to data linkage.

(1) Format

```

{{ extract authorization-identifier.table-identifier ( { column-name [ { { , column-
name } } ... ] | * } )
to update-information-name
key|ukey(column-name [ { { , column-name } } ... ] ) [ check
{ not_null_unique | unique | none } ] } } ...

```

(2) Explanation of the operands

■ extract

***authorization-identifier.table-identifier*({ *column-name* [{ { , *column-name* } } ...] | *)**

Specify the table that is to be subject to extraction and the columns to be extracted from that table. You must specify one `extract` statement for each extraction condition. You can specify only one *authorization-identifier.table-identifier* per `extract` statement.

To use multiple extraction conditions to extract a single table, specify a separate `extract` statement for each extraction condition. You can specify a maximum of 4000 column names per `extract` statement.

authorization-identifier.table-identifier

Specify the authorization identifier and table identifier of the table that contains the columns to be extracted. This must be a base table.

authorization-identifier

Specify the authorization identifier for the table that is to be subject to extraction of update information.

table-identifier

Specify the table identifier of the table that is to be subject to extraction of update information.

column-name

Specify the name of a column to be extracted. You can specify a maximum of 4000 columns. You can specify a repetition column only if no elements are specified, but array columns are not supported. You can specify the same column name more than once; however, if the column names are also specified in the `key` clause, they must be unique. The update information fields will reflect the order in which the columns were specified in the `extract` statement. An update information field is an area in the update information that is used to store an extracted HiRDB field.

#

Specify the asterisk (*) to have all columns in the specified table extracted as is without changing their order.

- **to update-information-name**

~ <symbolic name of 1-8 characters>

Specify a name for the extracted update information. Each update information name must be unique among all the extraction statements in the extraction definition.

- **{key|ukey}(column-name[{{,column-name }} ...])**

Specify the names of the columns to be used as the mapping key. Whether you use the `key` or `ukey` clause depends on whether the specified columns can be updated.

For details about the data types of columns that can be used as mapping keys, see *4.3.3(1) Mapping key when the source database is HiRDB*.

key column-name

Use the specified column as a mapping key column; this column cannot be updated.

ukey column-name

Use the specified column as a mapping key column; this column can be updated.

Note that if a mapping key is updated, the data linkage recovery facility cannot be used.

You can specify a maximum of 16 column names, but the same column name cannot be specified more than once. The specified column names must also be specified in the `extract` statement. If you specified `*` in the `extract` statement, you must specify one of the columns of the table that is to be subject to extraction processing.

■ **check {not_null_unique|unique|none}**

Specify the condition for performing a unique check on the mapping key column. The table below provides the details of unique checking. If the condition is not satisfied, extraction definition preprocessing results in a definition error. If this clause is omitted, Datareplicator performs checking based on the `-k` option specification in the `hdprep` command.

Table 5-7: Details of unique checking

Check item	Description
Index type	The index type must be one of the following: <ul style="list-style-type: none"> • Unique index • Unique cluster index • Primary index • Primary cluster index
Index component column	There must be only the mapping key component columns.

`not_null_unique`

Check the source table to confirm that the index satisfying the conditions described in Table 5-7 *Details of unique checking* has been defined and its index component columns have the NOT NULL attribute.

`unique`

Check the source table to confirm that the index satisfying the conditions described in Table 5-7 *Details of unique checking* has been defined. Because this option does not check the NULL value, you need to use a tool such as a user program that updates the source database to ensure the uniqueness of the data.

`none`

Do not perform a unique check. You need to use a tool such as a user program that updates the source database to ensure the uniqueness of the data.

(3) Notes

- If you specify the `key` clause, also specify columns that are not a mapping key in

the `extract` statement. Because mapping keys with the `key` clause specified are not updated, data cannot be extracted if only mapping-key columns are specified. If you specify the `ukey` clause, you can specify mapping-key columns only in the `extract` statement.

- You can extract null-value data.
- The authorization identifier, table identifier, column name, update information name, and target identifier specifications are case-sensitive.
- If you specify an authorization identifier, table identifier, column name, update information name, or target identifier that is the same as a source Datareplicator reserved word, you must enclose it in double quotation marks ("). For details about the source Datareplicator reserved words, see Appendix B. *Datareplicator Reserved Words*.
- A column name specified as part of the mapping key cannot be specified more than once in the `extract` statement.
- If you update the mapping key when the `key` clause is specified, the target system uses the updated data as the key. For this reason, an error might occur during import processing (no data with matching key) or data might be imported into an unexpected row. It is not advisable to update the mapping key.

5.5.4 Transmission definition statement

A transmission definition statement selects and sends update information extracted from the source database to a target identifier. The update information to be sent to a destination depends on how the transmission statements are specified. The following table shows the specification of transmission statements and the update information to be sent.

Table 5-8: Specification of transmission statements and the update information to be sent

Specification of transmission statements	Update information to be sent	
	Destination is specified in transmission statement	Destination is not specified in transmission statement
No transmission statements specified (omitted)	N/A	Update information for all update information names is sent to all destinations.
Transmission statements specified for some destinations	Update information for only those update information names specified in the transmission statements is sent.	Update information for all update information names is sent.
Transmission statements specified for all destinations	Same as above.	N/A

(1) Format

```
[ { { send target-identifier from update-information-name
  [ where column-name { relational-operator literal | in(literal [ , literal ] ... )
                        | flike(comparison-start-position , literal ) }
    [ and column-name { relational-operator literal | in(literal [ , literal ] ... )
                        | flike(comparison-start-position , literal ) } ] ]
  } } ... ]
```

(2) Explanation of the operands■ **send target-identifier**

~ <symbolic name of 1-8 characters>

Specify a target identifier that is to be subject to transmission of update information. This target identifier must be specified in the extraction system definition.

■ **from update-information-name**

~ <symbolic name of 1-8 characters>

Specify the name assigned to the update information that is to be subject to transmission to the target identifier specified in the `send` clause. This update information name must be specified in an extraction statement.

You can use right truncation to send to the same destination multiple units of update information, all of whose update information names begin with the same character string. Specify the common characters, followed by the percent sign (%); the percent sign can represent any number of characters, including no characters.

Example of right truncation

ABC%

Datereplicator sends update information for all update information names beginning with ABC (such as ABC, ABCAA, ABCABC).

■ **where clause**

By specifying a `where` clause, you can send only update information that satisfies specified conditions. The following explains the information that can be specified in the `where` clause:

- *column-name*

Specify the name of a mapping key in the update data subject to transmission. A mapping key column that is specified as a selection condition column must have one of the attributes listed in the table below. An error results if you specify a column that is not a mapping key column or a column whose attribute is not listed in this table.

Table 5-9: Attributes of the mapping key columns that can be specified in the selection conditions

Column attribute	Length, precision	Scaling	Permitted length
<code>char(n)</code>	$n \leq 255$	N/A	$1 \leq n \leq 255$
<code>mchar(n)</code>	$n \leq 255$	N/A	$1 \leq n \leq 255$
<code>nchar(n)</code>	$n \leq 127$	N/A	$1 \leq n \leq 127$
<code>varchar(n)</code>	$n \leq 255$	N/A	$1 \leq n \leq 255$
<code>mvarchar(n)</code>	$n \leq 255$	N/A	$1 \leq n \leq 255$
<code>nvarchar(n)</code>	$n \leq 127$	N/A	$1 \leq n \leq 127$
<code>[large]decimal(m,n)</code>	$1 \leq m \leq 38$	$0 \leq n \leq 38$	$1 \leq m \leq 38, m \leq n$
Integer	4 bytes	N/A	4 bytes
smallint	2 bytes	N/A	2 bytes

- *relational-operator*

Specify any of the following six relational operators:

=, <>, >, >=, <, <=

- *literal*

Specify a literal that is to be subject to comparison in the selection condition. Table 5-10 *Literals permitted in a selection condition* lists the literals permitted in selection conditions. Table 5-11 *Relationship between a literal and the attribute of the selection condition column* shows the relationship between a literal and the attribute of the selection condition column.

Table 5-10: Literals permitted in a selection condition

Type of literal	Description	Example
Character string literal	Character string enclosed in single quotation marks. To specify a single quotation mark in a character string literal, specify two consecutive single quotation marks, which will be recognized as one character of data. A literal's length must be 1 to 255 bytes.	'AB' 'CD'
Exact numeric literal or unsigned integer	Character string consisting of sign (+, -), number, and decimal point. You can specify a decimal number with a length of up to 29 digits.	-100 12.3

Table 5-11: Relationship between a literal and the attribute of the selection condition column

Column attribute	Literal	
	Character string literal	Exact numeric literal or unsigned integer
<code>char(n)</code>	Y ^{#1}	--
<code>mchar(n)</code>	Y ^{#1}	--
<code>nchar(n)</code>	Y ^{#1}	--
<code>varchar(n)</code>	Y ^{#2}	--
<code>mvarchar(n)</code>	Y ^{#2}	--
<code>nvarchar(n)</code>	Y ^{#2}	--
<code>[large]decimal(m,n)</code>	--	Y
<code>integer</code>	--	Y ^{#3}
<code>smallint</code>	--	Y ^{#4}

Y: Literal can be specified

--: Literal cannot be specified

#1: $n <$ length of character string literal: Definition error.

$n >$ length of character string literal: Datareplicator pads the character string literal with space characters to adjust its length to the column length, and then compares. For a column with a character set specification, Datareplicator pads the character string literal with the space characters of that column's character set. For a column with no character set specification, Datareplicator pads the character string literal with the space characters of the locale specified in the `dblocale` operand in the extraction system definition. For `nchar`, Datareplicator adds double-byte spaces; a definition error results if you specify an odd number of bytes for a character string literal for a column with the `nchar` attribute.

#2: $n <$ length of character string literal: Definition error.

$n >$ length of character string literal: Datareplicator compares the character string literal only from the left and, if they match, compares the rest of the character string.

#3: You can specify an integer in the range -2147483648 to 2147483647 . Specification of any other value will result in a definition error.

#4: You can specify an integer in the range -32768 to 32767. Specification of any other value will result in a definition error.

- `in(literal[, literal...])`

Specify a matching condition. Specify one or more literals such that the condition will be true if any of the specified literals matches the data in the specified selection condition column. You can specify a maximum of 16 literals in this condition. The specification for the literal specified with the relational operator also applies to these literals.

- `flike(comparison-start-position , literal)`

Specify a partial matching condition. The condition will be true if the character string beginning at the specified comparison start position in the selection condition column data matches the specified literal. The permitted range of comparison start positions is from 0 to 254. The specification for the literal specified with the relational operator also applies to this literal. However, Datareplicator adds no space characters to this literal during comparison.

If the definition length of the selection condition column is shorter than (comparison start position + defined length), a definition error results. If the selection condition column is a variable-length column and the length of the real data in the selection condition column is shorter than (comparison start position + defined length), the condition is false.

(3) Notes

- A definition error results if you specify more than one transmission statement containing the same target identifier and update information name.
- You must specify the transmission definition statements so that there is at least one destination for each update information name defined in the extraction statements. If no destination can be determined for update information, a definition error results.
- You can specify a maximum of 256 selection conditions in the `where` clause.
- You cannot specify the `where` clause if you use right truncation to specify the update information name in the `from` clause.
- You cannot specify the `where` clause if you specify the `from` clause to make multiple update information names from the same table subject to extraction (defining multiple update information names).
- If a column with a character set specification is specified for the selection condition in the `where` clause, the comparison results depend on the relationship among the characters in the character set specified for that column.
- If the selection condition column data is the `null` value, the condition will be

false.

- If the `ukey` clause is specified in the `extract` statement that defines the update information name specified in the `send` statement, Datareplicator uses pre-update data to check the conditions. If you update a mapping key column specified in a transmission condition to a value that does not satisfy the transmission condition, the destination data will be updated to a key that is not subject to transmission. In this case, the updated data becomes invalid and the correct data will not be found at the destination corresponding to the updated value.

To update a mapping key column specified in a transmission condition to a value outside the range of the transmission conditions, you must execute `DELETE` and `INSERT` to achieve data conformity instead of simply executing `UPDATE`.

5.6 Source HiRDB definition

To extract data from a HiRDB database, you must define in the source HiRDB definition information required to use the source Datareplicator. The table below shows the HiRDB definitions needed to use the source Datareplicator. For details about the format of the HiRDB definitions and other HiRDB-related definitions, see the *HiRDB Version 9 System Definition* or *HiRDB System Definition* manual.

Table 5-12: HiRDB definitions needed to use the source Datareplicator

Name of definition	Format	Operand	Description	Change
System common definition	set format	pd_rpl_init_start	Specifies the method for starting data linkage at the source HiRDB.	--
		pd_log_rpl_no_standby_file_opr	Specifies the handling of data linkage if system log file swapping fails.	Y
Unit control information definition	set format	pd_rpl_hdepath	Specifies the directory used at the source Datareplicator.	--

Y: Operand that can be changed when HiRDB is restarted.

--: Operand that cannot be changed when HiRDB is restarted.

5.6.1 System common definition

(1) Format

```

:
[ set pd_rpl_init_start=Y|N ]
[ set pd_log_rpl_no_standby_file_opr=stop|continue ]
:

```

(2) Explanation of the operands

■ pd_rpl_init_start=Y|N

Specify whether HiRDB Datareplicator linkage is to be started when HiRDB starts.

Y

Start HiRDB Datareplicator linkage when HiRDB starts. Output to the system log of data linkage information needed for HiRDB Datareplicator linkage starts at the time of HiRDB startup.

N

Do not start HiRDB Datareplicator linkage when HiRDB starts. You must use HiRDB's `pdrplstart` command to start HiRDB Datareplicator linkage. For details about how to specify the `pdrplstart` command, see 6.5 *Handling of the source HiRDB*.

■ **pd_log_rpl_no_standby_file_opr=stop|continue**

Specify the handling of HiRDB Datareplicator linkage if a swapping request is issued while none of the system log files can be swapped because extraction of system log information has not been completed at the source Datareplicator. This operand is applied only when HiRDB Datareplicator linkage is executed by specification of `pd_rpl_init_start=Y` or by execution of the `pdrplstart` command. This operand is not applicable to a HiRDB/Parallel Server's front-end server or dictionary server.

stop

Forcibly terminate the HiRDB unit. For details about how to restart a forcibly terminated HiRDB unit, see 6.5 *Handling of the source HiRDB*.

continue

Cancel HiRDB Datareplicator linkage and continue only HiRDB's processing. In this case, conformity between the source and target databases subject to data linkage is lost, so you must re-create the target database. For details about how to re-create a target database, see 6.5 *Handling of the source HiRDB*.

5.6.2 Unit control information definition

(1) Format

```

:
set pd_rpl_hdepath=source-Datareplicator-directory-name
:

```

(2) Explanation of the operand

■ **pd_rpl_hdepath=source-HiRDB-Datareplicator-directory-name**

~ <pathname>

Specify the directory name used by the source Datareplicator. This is the directory name specified in the source Datareplicator's `HDEPATH` environment variable.

If you omit this operand, you cannot use the HiRDB Datareplicator linkage facility. This operand is applied only when HiRDB Datareplicator linkage is executed by specification of `pd_rpl_init_start=Y` or by execution of the `pdrplstart` command in the system common definition.

5.7 Duplexing definition (source)

The duplexing definition specifies the correspondence between the logical and physical file names that are used in the source system's duplexing definition.

(1) Format

```
{ { logical_file = logical-file-name
  physical_file_a = physical-file-name-1
  physical_file_b = physical-file-name-2
# comment-line
} }
:
```

The length of a single definition statement line cannot exceed 1,024 single-byte characters. To specify a comment, start the line with a hash mark (#).

(2) Explanation of the operands

logical_file = logical-file-name

Specify the logical file name of the file to be duplexed. The following table shows the logical file name for each file:

System	Definition file	Logical file name
Source system	Extraction master status file	mststatus
	Extraction server status file	sts_server-name
	Extraction information queue file	name-specified-in-extraction-environment-definition_server-name
	Data linkage file	hde_server-name

If the same logical file name is defined more than once, a definition analysis error occurs. If the specified logical file name does not exist in the extraction environment definition file, all the definitions associated with that logical file name will be ignored.

physical_file_a = physical-file-name-1, physical_file_b = physical-file-name-2

5. Definitions

~ ((1-125)) (bytes)

Specify the absolute path names of the physical files that constitute the logical file specified in `logical_file`. If the same physical file name is defined more than once, a definition analysis error occurs. If you are using a character special file, create a symbolic link in such a manner that no identical name is used between nodes and make sure that all physical file names are unique.

5.8 Import system definition

An import system definition specifies information about a target Datareplicator's operating environment.

5.8.1 Format

```
[ set hdsid=target-Datareplicator-identifier ]
[ set hirdbusr=HiRDB-connection-authorization-identifier[ /password ] ]
[ set protocol1=tcp|osi ]#1
[ set protocol2=tcp|osi ]#1
  set dsid001=data-linkage-identifier
[... [ set dsid128=data-linkage-identifier ] ]#2
  set refenv001="import-environment-definition-filename"
[... [ set refenv128="import-environment-definition-filename" ] ]#2
[ set hdsservice=service-name ]
[ set reflect_tselector=T-selector ]#1
[ set hirdb_audit_trail=all|uoc|none_cont|none_stop ]
[ set keepalive=true|false ]
[ set errfilesz=import-error-information-file-size ]
[ set syslogout=true|false ]
[ set syslog_message_suppress=message-number[ ,message-number] ... ]
[ set dblocale={ sjis|euc|utf-8|unknown } ]
[ set msglocale={ english|sjis-japanese|euc-japanese } ]
[ set discintvl=disconnect-issuance-interval ]
[ set info_message_out=nosuppress|suppress ]
[ set except_suppress=message-number[ ,message-number] ... ]
[ set commitment_method=fxa_none|fxa_sqle ]
[ set int_trc_lvl=activity-trace-collection-interval[ ,activity-trace-collection-range ] ]
[ set int_trc_filesz=activity-trace-file-size ]
[ set int_trc_rintvl=activity-trace-information-collection-interval ]
[ set use_convertlib=true|false ]#3
[ set
ref_wait_interval=import-process's-import-information-queue-file-read-interval ]
[ set commit_wait_time=COMMIT-issuance-interval ]
[ set file_dupenv=duplexing-definition-file-name ]
[ set syncgroup001=
synchronous-import-group-name , data-linkage-identifier [ { { , data-linkage-identifier
}}... ] ]
[ set syncgrp_discintvl= disconnect-issuance-wait-time ]
[ set syncwait_limit_tran_count=
maximum-number-of-transactions-to-wait-for-until-synchronization ]
[ set syncwait_limit_time=maximum-time-to-wait-until synchronization ]
[ set reflect_counter_reset=true|false ]
```

```
[ set resource_chk_err=continue|stop ]
```

#1

Specify this operand if you use the OSI protocol for communication. This operand is applicable only to the HP-UX edition of Datareplicator (excluding the IPF version).

#2

If you are using the Windows Datareplicator, do not specify the `dsid064` to `dsid128` or `rfenv064` to `rfenv128` operands because this Datareplicator supports only 63 target identifiers and import environment definition files. Specifying these operands will result in an error.

#3

The `use_convertlib` operand is applicable only to Windows Datareplicator; do not specify this operand for UNIX Datareplicator.

5.8.2 Modifying defined information

To modify defined information:

1. Terminate the target Datareplicator.

To execute an initial start subsequently, the termination mode must be normal, immediate, or event termination. If the target Datareplicator has terminated abnormally, start it normally, and then perform the procedure to modify the defined information.

2. Use a text editor to modify the defined information.
3. Start the system in a start mode that will apply the modified information, as shown in the following table.

Table 5-13: Start mode that applies the modified import system definition

Operand name	Start mode that applies the modified information		
	Initial start ^{#1}	Partial initial start ^{#1}	Normal start ^{#1}
<code>hdsid</code>	Y	--	--
<code>hirdbusr</code>	Y	Y	Y
<code>protocol1</code>	Y	--	--
<code>protocol2</code>	Y	--	--
<code>dsidxx</code> ^{#2}	Y	Y	--
<code>refenvxx</code> ^{#2}	Y	Y	--

Operand name	Start mode that applies the modified information		
	Initial start ^{#1}	Partial initial start ^{#1}	Normal start ^{#1}
hdsservice	Y	Y	Y
reflect_tselector	Y	Y	Y
hirdb_audit_trail	Y	Y	Y
keepalive	Y	Y	Y
errfilesz	Y	Y	Y
syslogout	Y	Y	Y
syslog_message_suppress	Y	Y	Y
dblocale	Y	--	--
msglocale	Y	Y	Y
discintvl	Y	Y	Y
info_message_out	Y	Y	Y
except_suppress	Y	Y	Y
commitment_method	Y	--	--
int_trc_lvl	Y	Y	Y
int_trc_filesz	Y	Y	Y
int_trc_rintvl	Y	Y	Y
use_convertlib	Y	Y	Y
ref_wait_interval	Y	Y	Y
commit_wait_time	Y	Y	Y
file_dupenv	Y	--	--
syncgroup001	Y	--	--
syncgrp_discintvl	Y	Y	Y
syncwait_limit_tran_count	Y	Y	Y
syncwait_limit_time	Y	Y	Y
reflect_counter_reset	Y	Y	Y

Operand name	Start mode that applies the modified information		
	Initial start ^{#1}	Partial initial start ^{#1}	Normal start ^{#1}
resource_chk_err	Y	Y	Y

Legend:

Y: The start mode denoted by this column applies the operand's modified information. If the start modes of multiple columns apply a particular operand's modified information, any one of those start modes can be used to apply the modified information.

--: Not applicable

#1

This is the start mode used to start the target Datareplicator.

#2

xxx is a value in the range from 001 to 128. If you are executing normal start, do not change any of the dsid001 to dsid128 operands or refenv001 to refenv128 operands.

5.8.3 Explanation of the operands

- **hdsid=source-Datareplicator-identifier**

~ <hexadecimal> ((00-FF)) <<00>>

If there are multiple target Datareplicators, specify the identifier of a target Datareplicator. You can omit this operand if there is only one target Datareplicator.

- **hirdbusr=HiRDB-connection-authorization-identifier[/password]**

<<value of the target HiRDB's PDUSER environment variable>>

Specify the authorization identifier and password to be used to establish connection with the target HiRDB.

- *HiRDB-connection-authorization-identifier* ~ <identifier of 1-8 characters>

Specify the authorization identifier to be used to establish connection with the target HiRDB. An attempt to establish connection by specifying an invalid authorization identifier will result in an error.

- */password* ~ <symbolic name of 1-28 character>

Specify the password to be used to establish connection with the target HiRDB. An attempt to establish connection by specifying an invalid password will result in an error. If you omit */password*, Datareplicator will

establish connection with the target HiRDB without requiring a password.

In the UNIX edition, if you specify the asterisk (*) for the password, the target Datareplicator will issue to the standard input during startup a request for entry of a password; the entered password will be used to establish connection with the target HiRDB.

In the Windows edition, if you specify the asterisk (*) for the password, Datareplicator will establish connection with the target HiRDB without requiring a password.

It is not appropriate to enclose this operand's value in double quotation marks in order to make it case sensitive. When a value is enclosed in double quotation marks, an error results because each double quotation mark is handled as a part of the user ID or password.

■ **protocol1=tcp|osi**

Specify the communications protocol to be used with the source system.

tcp

Use the TCP/IP protocol for communication.

osi

Use the OSI protocol for communication.

This operand is applicable only to the HP-UX edition of Datareplicator (excluding the IPF version).

■ **protocol2=tcp|osi**

Specify a second communications protocol to be used with the source system. Datareplicator ignores this operand if the specified protocol is the same protocol specified in the `protocol1` operand.

tcp

Use the TCP/IP protocol for communications.

osi

Use the OSI protocol for communications.

This operand is applicable only to the HP-UX edition of Datareplicator (excluding the IPF version).

■ **dsid001=data-linkage-identifier... dsid128=data-linkage-identifier**

~ <hexadecimal ((00-FF)) or **>

Specify the source system data linkage identifiers that are to be subject to import processing. These data linkage identifiers must be unique in the import system

definition.

If multiple data linkage identifiers are subject to data linkage, specify the `dsid001` to `dsid128` operands consecutively in ascending order beginning with `dsid001`. If they are not consecutive or in ascending order, Datareplicator uses only those operands that are specified consecutively in ascending order from the beginning.

When you specify `**` in place of a data linkage identifier, Datareplicator assumes an absent number, in which case no data linkage occurs (no startup of a reception process or definition server process). You can specify `**` in place of any number of data linkage identifiers; if you specify `**` for all data linkage identifiers, Datareplicator assumes that no data linkage identifiers at all are specified.

In the case of the Windows Datareplicator, you can specify a maximum of 63 data linkage identifiers corresponding to import environment definition files. You cannot specify any of the `dsid064` to `dsid128` operands; specifying these operands will result in an error.

- **refenv001="import-environment-definition-filename" ...
refenv128="import-environment-definition-filename"**

~ <[*pathname* /]*filename* of 1-64 bytes>

Specify the filenames of the import environment definitions that define operating environments for import processing, as absolute or relative pathnames. If you specify a relative pathname, Datareplicator assumes `$HDSPATH/relative-pathname` as the absolute pathname. In this case, specify the pathname so that the final absolute pathname will not exceed 125 characters.

You must specify the `refenv001` to `refenv128` operands so that they correspond to the `dsid001` to `dsid128` operands. If there is no `refenvxxx` operand corresponding to a `dsidxxx` operand, an error results. On the other hand, if there is a `refenvxxx` operand but no corresponding `dsidxxx` operand, Datareplicator ignores that `refenvxxx` operand.

In the case of Windows Datareplicator, you can specify a maximum of 63 data linkage identifiers and corresponding import environment definition files. You cannot specify any of the `refenv064` to `refenv128` operands; specifying any of these operands will result in an error.

- **hdsservice=service-name**

~ <identifier of 1-64 characters> <<hirdbds>>

Specify the service name that was used to add the target Datareplicator's communication entry to the `services` file.

- **reflect_tselector=T-selector**

~ <even number of hexadecimal digits with a length of 2-64 digits>

This operand is required if you specify `osi` in the `protocol1` or `protocol2` operand. Specify the T-selector assigned to the source and target Datareplicators. You must specify the same T-selector at both the source and target systems. If the T-selector specified at the source system is a character string, specify its hexadecimal representation. This operand is applicable only to the HP-UX edition of Datareplicator (excluding the IPF version).

■ **`hirdb_audit_trail=all | uoc | none_cont | none_stop`**

Specify whether an audit trail is to be collected if an operation subject to audit trail collection (audit event) occurs.

The import definition server process, import SQL process, and import UOC processes other than the `CONNECT` process are subject to collection of audit events. For import UOC processes, this operand is applicable only to an import information editing UOC routine that accesses `HiRDB`. If you use an import information editing UOC routine but do not collect an audit trail, make sure that you see *8.1.6(7) Notes when not collecting an import processing-related audit trail*.

`all`

Collect an audit trail on the import Datareplicator's processing.

`uoc`

Collect an audit trail on the import information editing UOC routine's processing.

`none_cont`

Do not collect an audit trail. If the `KFRB00864-W` or `KFRB03094-W` message is issued from an import information editing UOC routine, resume processing with audit trail collection activated.

`none_stop`

Do not collect an audit trail. If the `KFRB00865-E` or `KFRB03094-W` message is issued from an import information editing UOC routine, stop the processing.

If you want to minimize the overhead of having import Datareplicator processing collect an audit trail, we recommend that you specify `none_cont` or `none_stop`.

■ **`keepalive=true|false`**

Specify whether the `keepalive` option can be specified for the socket.

`true`

Specify the `keepalive` option.

`false`

Do not specify the `keepalive` option.

The `keepalive` operand is not applicable when `osi` is specified in the `protocol1` or `protocol2` operand.

■ **`errfilesz=import-error-information-file-size`**

~ <unsigned integer> ((1-32767)) <<16>> (KB)

Specify the maximum size of an import error information file.

■ **`syslogout=true|false`**

Specify whether the information in the import error information files is to be output also to the syslog file. If you are using JP1, you must output this information to the syslog file to implement automatic operation.

`true`

Output the information in the import error information files also to the syslog file.

`false`

Do not output the information in the import error information files to the syslog file.

Note that the `KFRB00501-I` and `FRB00505-I` messages are still output to the syslog file even when `false` is specified.

■ **`syslog_message_suppress=message-number[,message-number] ...`**

Specify the numbers of the messages whose output to the syslog file (event log for Windows) is to be suppressed.

- Specify a maximum of 64 message numbers.
- If you specify the same message number more than once, only the first specification of the number is effective.
- Datareplicator does not output any message specified in this operand, regardless of its significance level (E, W, I, or Q).
- Messages that are output only to the syslog file (or event log) are always output regardless of whether you specify this operand.

■ **`dblocale={ sjis|euc|utf-8|unknown }`**

Specify the character code system to be used when update information sent from the source system is converted to the target system's storage character codes. During analysis of the import definition, Datareplicator assumes that the import definition is coded in the code system specified in this operand.

`sjis`

Use the JIS8/Shift JIS code system.

`euc`

Use the EUC code system.

`utf-8`

Convert data to the utf-8 character code system.

`unknown`

Do not change the character code system. You specify this option when a UOC routine will be used to convert character codes. During analysis of the import definition when `unknown` is specified, Datareplicator interprets the import definition on the basis of the specification of the `LANG` environment variable, as follows:

`LANG=ja_JP.SJIS`: JIS8/Shift JIS code system

`LANG=other`: EUC code system

The default value for this operand depends on the OS in use. Check the following table to determine the default value:

OS	Default value
HP-UX	<code>sjis</code>
AIX	
Windows	
Solaris	<code>euc</code>
Linux	

In the case of Windows Datareplicator, you must specify `sjis`, `utf-8`, or `unknown`, because the source Datareplicator supports only the JIS8/Shift JIS and Unicode systems. Specification of any other value in the `dblocate` operand will result in an error.

■ **`msglocale={ english|sjis-japanese|euc-japanese }`**

Specify the character code set to be used for messages issued by the source Datareplicator.

For Windows Datareplicator, specify either `english` or `sjis-japanese`; specifying `euc-japanese` will result in an error.

`english`

Output messages in English.

`sjis-japanese`

Output messages in Japanese using the JIS8/Shift JIS code system.

`euc-japanese`

Output messages in Japanese using the EUC code system. This option is not available to Windows Datareplicator.

■ **`discintvl=disconnect-issuance-interval`**

~ <unsigned integer> ((0-65535)) <<180>> (seconds)

Specify the interval after which the target Datareplicator is to issue `disconnect` to the target HiRDB once it detects the end of the update information in the import information queue file during import processing. If you specify 0, Datareplicator does not issue `disconnect` to the target HiRDB when the end of the update information is detected during import processing. The `disconnect` issuance timing is the same as with the `commit_wait_time` operand.

■ **`info_message_out=nosuppress|suppress`**

Specify whether output of information-only messages to the syslog file (event log for Windows) and error information files is to be suppressed.

`nosuppress`

Do not suppress output of information-only messages.

`suppress`

Suppress output of information-only messages. The following message numbers are subject to this operand's specification:

00100, 00103, 00104, 02019, 02020, 02021, 02022, 03001, 03002,
03008, 03009, 03011, 03012, 03013, 03022, 03201, 03202, 03204,
03209, 03028, 03058, 03301, 03302

■ **`except_suppress=message-number[,message-number] ...`**

~ <5-digit unsigned integer>

From among the messages whose output is suppressed because `info_message_out=suppress` is specified, specify the numbers of the messages whose output suppression is to be cancelled and which are to be output to the syslog and error information files. You can specify a maximum of 63 message numbers.

Datareplicator ignores specification of a message number if it is not subject to output suppression (specifying such a message number serves no purposes). If you specify the same message number more than once, only the first specification

is effective.

This operand is applicable only when `suppress` is specified in the `info_message_out` operand.

■ **`commit_method=fxa_none|fxa_sqle`**

Specify the synchronization point processing method for import processing (single-phase commit method or double-phase commit method). For details about the synchronization point processing method for import processing, see 3.3.12 *Specifying the synchronization point processing for import processing*.

When you have changed this operand, execute an initial start of the target Datareplicator.

`fxa_none`

Use the single-phase commit method to execute synchronization point processing. If the target RDBMS is not HiRDB, Datareplicator assumes this value and executes synchronization point processing using the single-phase commit method. To use the import information editing UOC routine, specify `fxa_none`.

`fxa_sqle`

Use the double-phase commit method to execute synchronization point processing with the import SQL process. However, note that the import UOC process uses the single-phase commit method to execute synchronization point processing.

If you use the import transaction synchronization facility (`syncgroup001` operand specified), specify `fxa_sqle`.

Also specify `fxa_sqle` if the target columns include repetition columns.

If you are using multiple target Datareplicators for a single HiRDB to import data using the double-phase commit method, the target Datareplicator identifiers cannot be the same. If data is imported into a single HiRDB, synchronization point processing cannot be executed correctly.

Important

If you specify `fxa_sqle` in this operand, make sure before you initialize the target Datareplicator that no undetermined transactions remain in the target HiRDB.

If any undetermined transactions remain, first use the `pdcmr` or `pdrbk` command to determine the transactions, and then initialize the target Datareplicator.

■ **`int_trc_lvl=activity-trace-collection-level[,activity-trace-collection-range]`**

To change the items to be collected in the activity trace files (import trace files), specify the appropriate value shown in the table below. If you omit this operand, Datareplicator collects only the information common to all facilities (the minimum requirement).

If you specify `na` for the activity trace collection level, Datareplicator will not collect activity trace information and will ignore any value specified for the activity trace collection range (however, Datareplicator still checks for syntax and range errors).

- Value for the activity trace collection level

Value	Information to be collected		
	Common information	Overview of performance	Details about performance
<code>p1</code>	Y	Y	--
<code>p2</code>	Y	Y	Y
<code>na</code>	--	--	--
<code>int_trc_lvl</code> operand omitted	Y	--	--

Y: Collected.

--: Not collected.

Common information: Collective name for the general checkpoint information that indicates a point of change in start/termination information, error information, or process level.

Note:

When you specify `p2`, you must provide sufficient space for the activity trace files. Otherwise, important information might be deleted by the wraparound feature; in addition, trace collection might result in a large overhead workload.

- Value for the activity trace collection range

Value	Information to be collected			
	MST (control)	RCV (reception)	RFC (import)	SQE (SQL execution)
<code>c1</code>	Y	Y	Y	--
<code>c2</code>	Y	Y	--	Y
<code>c3</code>	Y	Y	--	--

Value	Information to be collected			
	MST (control)	RCV (reception)	RFC (import)	SQE (SQL execution)
c4	Y	--	Y	Y
c5	Y	--	Y	--
c6	Y	--	--	Y
c7	Y	--	--	--
nc	Y	#	#	#
<i>int_trc_lvl</i> operand or <i>activity-trace-collection-range</i> omitted	Y	Y	Y	Y

Y: Collected.

--: Not collected.

#: Even if you specify *nc* in the import system definition, you can still collect activity trace information individually by specifying the *int_trc_get1* operand in the import environment definition.

Guidelines for the *int_trc_lvl* operand specification:

We recommend that you specify the *int_trc_lvl* operand as follows:

1. Production run

For actual operations, we recommend that you omit the *int_trc_lvl* operand. If you cannot obtain reasonable performance with specification of the operand omitted, specify collection of activity trace information temporarily so that you can evaluate performance. In this case, specify *p1* or *p2* in the first parameter of the *int_trc_lvl* operand, *nc* in the second parameter, and the *int_trc_get1* or *int_trc_getv* operand in the specific facility (the one exhibiting low performance). If you specify *p2* in the first parameter of the *int_trc_lvl* operand, specify a sufficient value in the *int_trc_filesz* operand.

2. Test run

At the test stage, we recommend that you specify *p1* in the first parameter of the *int_trc_lvl* operand and omit the second parameter. This enables you to execute the *hdstrcdit* command to view activity trace information in the event of a problem with performance. Use the command's execution results to tune the HiRDB table definitions and the number of import groups.

If you need more detailed information (such as the unit price of SQL

executions), change the second parameter to `p2` in the `int_trc_lvl` operand. When you specify `p2`, specify a sufficient value in the `int_trc_filesz` operand (at least 1MB).

■ **int_trc_filesz=activity-trace-file-size**

~ <unsigned integer> ((32-1048576)) <<128>> (KB)

Specify the maximum storage size for an activity trace file (import trace file). Datareplicator ignores this operand when `na` is specified in the `int_trc_lvl` operand.

We recommend that you specify a value that is a multiple of 32. Otherwise, Datareplicator rounds up the specified value to the nearest multiple of 32KB and uses that value as the maximum size for an activity trace file.

In the target system, Datareplicator creates the activity trace files under the following names in the `$HDSPATH` directory (Datareplicator uses dual files with swapping and wrapping):

Import trace files

`$HDSPATH/reftrc.trc1` and `$HDSPATH/reftrc.trc2`

■ **int_trc_rintvl=activity-trace-information-collection-interval**

~ <unsigned integer> ((5-30000)) <<50>> (milliseconds)

Specify the interval at which activity trace information is to be collected. If `na` is specified in the `int_trc_lvl` operand, Datareplicator ignores this operand if it is specified.

You can minimize the amount of missed activity trace information by specifying a short activity trace information collection interval (specify a small value), but the number of monitorings per second increases, resulting in a higher CPU utilization factor. Specify a small activity trace information collection interval in the following cases:

- There are missing entries in the activity trace information.
- The specified activity trace collection level indicates a high level of collection information.
- The frequency of Datareplicator activity is high (there is a large amount of import data).

On the other hand, specify a large activity trace information collection interval in the following case:

- The frequency of Datareplicator activity is low (there is a small amount of import data) and a reduction of CPU utilization by Datareplicator is desired.

■ **use_convertlib=true|false**

Specify whether the code conversion facility is to be used to convert character codes.

`true`

Use the code conversion facility to convert character codes. This value is applicable only when the following character encoding is used at the source database:

- EBCDIK/KEIS78
- EBCDIK/KEIS83
- SJIS

If the code conversion facility is not installed, Datareplicator ignores `true` if it is specified.

`false`

Do not use the code conversion facility to convert character codes. In this case, all Gaiji codes will be converted to spaces. To convert Gaiji characters, use the `hdscnvedt` command to change the Gaiji conversion method.

For details about the `hdscnvedt` command, see the *hdscnvedt* command in 7. *Command Syntax*.

The `use_convertlib` operand is applicable only to Windows Datareplicator. Do not specify this operand for UNIX Datareplicator.

■ **ref_wait_interval=import-process's-import-information-queue-file-read-interval**

~ <unsigned integer> ((100-60000)) <<5000>> (milliseconds)

Specify the interval after which Datareplicator is to restart the next read operation once the import process detects the end of the import information queue file.

By specifying a small value in this operand, you can improve the spontaneity of import processing because of the brief wait time after the end of the import information queue file is detected. However, this might have an adverse effect on performance because the CPU utilization factor increases. You must take into account the frequency of update processing and CPU performance in specifying this value.

■ **commit_wait_time=COMMIT-issuance-interval**

~ <unsigned integer> ((0-300)) <<30>> (seconds)

Specify the interval after which Datareplicator is to issue `COMMIT` to the target HiRDB once the import process detects the end of the import information queue file. If you specify 0, Datareplicator issues `COMMIT` as soon as the end of the import information queue file is detected. Whether you specify or omit this

operand, Datareplicator reads the import information queue file at the interval specified in the `ref_wait_interval` operand; it issues a `COMMIT` on the basis of the passage of the amount of time specified in this operand. Therefore, depending on the `ref_wait_interval` operand's value, `COMMIT` might not be issued exactly as specified by this operand.

If the end of the import information queue file is detected but no update information has been stored in the import information queue file since the previous issuance of `COMMIT`, Datareplicator does not issue a `COMMIT` even though the amount of time specified with this operand has elapsed.

If a large value is specified in this operand, the longer will be the interval from the detection of the end of the import information queue file to the issuance of `COMMIT`, and, except for the time the search with no lock is conducted, there will be less time to reference data.

If a smaller value is specified in this operand, the shorter will be the interval from the detection to the issuance of `COMMIT`, so there will be more time to reference the data. However, depending on the specified transmission interval, Datareplicator might need to receive and import the next data immediately after issuing a `COMMIT`.

As a result, Datareplicator issues `COMMIT` more frequently, thereby reducing throughput. Therefore, if the transmission interval is less than one minute, we recommend that you specify half the transmission interval in this operand; if the transmission interval is one minute or more, we recommend that you specify the default value. If the transmission interval is 0, specify this operand taking into account the average of the actual transmission intervals.

If you increase the value of this operand, check that the value of the environment variable associated with HiRDB client monitoring is still greater than this operand's value. If this operand's value becomes greater than the `discintvl` operand's value, Datareplicator issues a `COMMIT` when the `discintvl` operand's value is reached, in which case specifying this operand serves no purposes. When you specify this operand, check that its value is smaller than the `discintvl` operand's value. Even if this operand's value is greater than the `discintvl` operand's value, Datareplicator will not detect an error.

■ **file_dupenv=duplexing-definition-file-name**

~ <filename of 1 to 125 bytes>

Specify the absolute or relative path name of the duplexing definition file. If a relative path is specified, the system assumes that the path is relative to the target Datareplicator directory.

If the `file_dupenv` operand is omitted, Datareplicator assumes that the duplexing function is not used.

- **syncgroup001=synchronous-import-group-name,data-linkage-identifier[,{data-linkage-identifier}]...**

Specify this operand in order to use the import transaction synchronization facility.

When you specify this operand, specify `fxa_sqlc` in the `commit_method` operand (sets the two-phase commit method as the synchronization point processing method for import processing).

synchronous-import-group-name

~ <symbolic name of 1-8 characters>

Specify the name to be assigned to a synchronous import group.

data-linkage-identifier

Specify a data linkage identifier that was specified in the `dsidxxx` operand. An error results if a specified data linkage identifier is not specified in the `dsidxxx` operand or if `**` is specified.

Each specified data linkage identifier must be unique.

The maximum number of data linkage identifiers that can be specified per synchronous import group is 128 for UNIX Datareplicator and 63 for Windows Datareplicator. For Windows Datareplicator, operands `dsid064` through `dsid128` cannot be specified; specifying any of these operands results in an error.

Regarding the data linkage identifiers specified in this operand, there are limitations to some of the operands. For details, see *3.7.5 Synchronous import group*.

- **syncgrp_discintvl=disconnect-issuance-wait-time**

<unsigned integer> ((0-65535)) <<180>> (seconds)

Specify the wait time before `disconnect` is issued when the import transaction synchronization facility is used.

If there is no update information to be imported before the specified wait time elapses after `commit` was issued, Datareplicator issues `disconnect` to the target HiRDB.

If a value of 0 is specified, Datareplicator will not issue `disconnect` to the target HiRDB even when the end of update information is detected during import processing.

This operand is applicable only to the data linkage identifiers that are specified for the synchronous import group.

- **syncwait_limit_tran_count=maximum-number-of-transactions-to-wait-for-u**

ntil-synchronization

<unsigned integer> ((2-65535)) <<4096>>

Specify the number of extraction transactions that can be processed as a single import transaction by each import process with the data linkage identifiers specified in the synchronous import group.

This operand value must be small enough for the target HiRDB's system log file even when the import transaction rolls back, and the number of locked resources used by the import transaction must be within the target HiRDB's lock pool size.

If no synchronous event is executed within the operand value range, the KFRB03303-E message is output and the synchronous import processing terminates with an error.

- **syncwait_limit_time=maximum-time-to-wait-until synchronization**

<unsigned integer> ((1-65535)) <<180>> (seconds)

For each import process of the data linkage identifiers specified in the synchronous import group, this operand specifies the maximum amount of time from when the end of the import information queue file is detected prior to detection of a synchronous event during import transaction processing, until the time at which the next update information is received.

This operand's value must be greater than the interval in which the `hdeevent` command is executed at the source HiRDB.

If no synchronous event is executed within the time specified in this operand, the KFRB03303-E message is output and the synchronous import processing terminates with an error.

- **reflect_counter_reset= true|false**

Specify whether the import processing count is to be reset when the target system starts.

`true`

Reset the import processing count when the target system starts.

`false`

Do not reset the import processing count when the target system starts.

Note:

If the import method used during restart of the target Datareplicator differs from the method used during the previous termination, the import processing count is reset even if `false` is specified. The following table shows whether the count is reset depending on the import method used during the previous termination and the import method used during restart of the target system:

Import method used during previous termination	Import method used during restart	Whether the count is reset
Transaction-based	Transaction-based	N
	Table-based	Y
Table-based	Transaction-based	Y
	Table-based	N [#]

Legend:

Y: Reset

N: Not reset

#: If there is a difference in any of the following items between the previous termination and the restart of Datareplicator, the import processing count is reset:

- Import group name
- Number of SQL processes
- Partitioning method (key range partitioning, hash partitioning)
- For key range partitioning, the number of key range groups
- For hash partitioning, the number of RDAREAs for the hash row-partitioned table

The following are the conditions under which the import processing count is reset other than during restart:

- The import method was changed by an event.
- The definition information was changed, and then the import process was restarted.

■ **resource_chk_err=continue | stop**

Specify the action to be taken if an error is detected when the target Datareplicator checks file integrity and file sizes during initialization or start processing.

If you always perform initialization or start processing when no change has been made to the data linkage environment or operands, we recommend that you specify `stop`.

`continue`

Continue with the initialization or start processing if it's possible for data linkage to be resumed or if the user changed operands intentionally.

5. Definitions

stop

Stop the initialization or start processing regardless of the nature of the error.

5.9 Import environment definition

An import environment definition specifies information needed to execute import processing. If multiple data linkage identifiers are subject to data linkage, Datareplicator executes import processing for each data linkage, so you must provide an import environment definition for each data linkage identifier.

5.9.1 Format

```

set qufile001="import-information-queue-filename"
set qufile002="import-information-queue-filename"
[... [ set qufile008="import-information-queue-filename" ]]
set queuesize=import-information-queue-file-size
[ set reffile="import-definition-filename" ]
set statsfile="import-status-filename"
set statssize=import-status-file-size
set unreffile1="unimported-information-filename-(primary)"
set unreffile2="unimported-information-filename-(secondary)"
[ set unreffilesz=unimported-information-file-size ]
[ set startmode={ trn|tbl|spd } ]
[ set restartmode=initial|continue ]
[ set breaktime=hh:mm ]
[ set breakmode=trn|tbl ]
[ set eventtrn=transaction-based-import-event-code ]
[ set eventtbl=table-based-import-event-code ]
[ set eventretrn=transaction-based-import-restart-event-code ]
[ set eventrettbl=table-based-import-restart-event-code ]
[ set eventspd=import-processing-stop-event-code ]
[ set eventcntreset=import-processing-count-reset-event-code ]
[ set defmerge=true|false ]
[ set cmtintvl=import-processing-commit-interval ]
[ set trncmtintvl=import-processing-commit-interval-for-transaction-based-import-
method ]
[ set tblcmtintvl=import-processing-commit-interval-for-table-based-import-method
]
[ set tblcheck=true|false ]
set defshmsize=shared-memory-size-for-storing-definition-information
[ set ebcDic_type={ eck78|ekk78|eck83|ekk83 } ]
[ set shiftspace_cnv=multi|single ]
[ set undefcode_cnv=multi|single ]
[ set ref_data_backspace=suppress|nosuppress ]
[ set skip_sqlcode=SQLCODE[, SQLCODE] ... ]
[ set skip_mvcelmwarn=true|false ]
[ set sqlerr_skip_info={ output|msgoutput|sqloutput|nooutput } ]
[ set extract_init=check|nocheck ]
[ set db_connect_retry_number=DB-connect-retries-count ]
[ set db_connect_retry_interval=DB-connect-retry-interval ]

```

```

[ set skip_codecvt_error=true|false ]
[ set int_trc_get1=activity-trace-collection-range ]
[ set device01=Datareplicator-file-system-area-name[ [ ,allocation-file-type] ... ]
[ set device02=Datareplicator-file-system-area-name[ [ ,allocation-file-type] ... ]
  [... [ set device09=Datareplicator-file-system-area-name[ [ ,allocation-file-
type] ... ]]]]
[ set ujcodekind=rcv|sam ]
[ set discintvl=disconnect-issuance-interval ]
[ set ref_wait_interval=
import-process's-import-information-queue-file-read-interval ]
[ set commit_wait_time=COMMIT-issuance-interval ]
[ set mapping_key_check={ not_null_unique|unique|none } ]
[ set control_trigger=execute|not_execute ]
[ set control_reference_trigger=execute | not_execute ]
[ set check_pending=use | nouse ]
[ set reflect_delay_limit_time=import-delay-period-threshold ]
[ set sql_lockerr_retrinum=
transaction-retry-count-in-the-event-of-a-lock-error ]
[ set xa_recovery_retry_count=transaction-recovery-request-retries-count ]
[ set xa_recovery_retry_interval=transaction-recovery-request-retry-interval
]
[ set reflect_trn_max_sqlnum=
maximum-update-SQL-statements-count-in-import-transaction ]

```

5.9.2 Modifying defined information

To modify defined information:

1. Terminate the import processing or the target Datareplicator.
If the target Datareplicator has terminated abnormally and an initial start is to be executed subsequently, start it normally, and then modify the defined information.
2. Use a text editor to modify the defined information.
3. Start the system in a start mode that will apply the modified information, as shown in the following table.

Table 5-14: Start mode that applies the modified import environment definition

Operand name	Start mode that applies the modified information		
	Initial start or partial initial start ^{#1}	Normal start ^{#1}	Import processing start ^{#2}
qufilexx ^{#3}	Y	--	--
queuesize	Y	--	--
reffile	Y	Y	Y

Operand name	Start mode that applies the modified information		
	Initial start or partial initial start ^{#1}	Normal start ^{#1}	Import processing start ^{#2}
statsfile ^{#4}	Y	--	--
statssize	Y	--	--
unreffile1	Y	--	--
unreffile2	Y	--	--
unreffilesz	Y	Y	--
startmode ^{#5}	Y	Y	--
restartmode ^{#5}	Y	Y	--
breaktime ^{#5}	Y	Y	--
breakmode ^{#5}	Y	Y	--
eventtrn	Y	--	--
eventtbl	Y	--	--
eventretrn	Y	--	--
eventretbl	Y	--	--
eventcntreset	Y	Y	--
eventspd	Y	--	--
defmerge	Y	Y	Y
cmtintvl	Y	Y	Y
trncmtintvl	Y	Y	Y
tblcmtintvl	Y	Y	Y
tblcheck	Y	Y	Y
defshmsize	Y	Y	Y
ebcdic_type	Y	Y	Y
shiftspace_cnv	Y	Y	Y
undefcode_cnv	Y	Y	Y

5. Definitions

Operand name	Start mode that applies the modified information		
	Initial start or partial initial start ^{#1}	Normal start ^{#1}	Import processing start ^{#2}
ref_data_backspace	Y	Y	Y
skip_sqlcode	Y	Y	Y
skip_mvcelmwarn	Y	Y	Y
sqlerr_skip_info	Y	Y	Y
extract_init	Y	Y	--
db_connect_retry_number	Y	Y	Y
db_connect_retry_interval	Y	Y	Y
skip_codecvt_error	Y	Y	Y
int_trc_get1	Y	Y	--
device ^{#6}	Y	--	--
ujcodekind	Y	--	--
discintvl	Y	Y	Y
ref_wait_interval	Y	Y	Y
commit_wait_time	Y	Y	Y
mapping_key_check	Y	Y	Y
control_trigger	Y	--	--
control_reference_trigger	Y	--	--
check_pending	Y	--	--
reflect_delay_limit_time	Y	Y	Y
sql_lockerr_retrypnum	Y	Y	Y
xa_recovery_retry_count	Y	Y	Y
xa_recovery_retry_interval	Y	Y	Y
reflect_trn_max_sqlnum	Y	Y	Y

Legend:

Y: The start mode denoted by this column applies the operand's modified information. If the start modes of multiple columns apply a particular operand's modified information, any one of those start modes can be used to apply the modified information.

--: Not applicable

#1

This is the start mode used to start the target Datareplicator.

#2

This is the start mode used to start the target Datareplicator's import processing.

#3

xxx is a value in the range from 001 to 008.

#4

If you are executing a normal start, do not change the `statsfile` operand.

#5

If the target Datareplicator terminated with an error or in the forced termination mode, it inherits the operand settings in effect at the time of the previous termination in order to maintain conformity between the source and target databases.

#6

xx is a value in the range from 01 to 09.

5.9.3 Explanation of the operands

- **qufile001='import-information-queue-filename'...qufile008='import-information-queue-filename'**

~ <[*pathname*/]*filename* of 1-64 bytes>

Specify the names of the import information queue files, as absolute or relative pathnames. If you specify a relative pathname, Datareplicator assumes `$(HDSPPATH)/relative-pathname` as the absolute pathname. In this case, specify the pathname so that the final absolute pathname will not exceed 125 characters.

Datareplicator uses file swapping to store information in the import information queue files. Therefore, you must provide at least two import information queue files; the maximum is eight files. Do not neglect to specify at least `qufile001` and `qufile002`.

You must specify the `qufile001` to `qufile008` operands consecutively in

ascending order beginning with `qufile001`. If they are not consecutive or in ascending order, Datareplicator uses only those import information queue files that are specified consecutively in ascending order from the beginning. Each filename must be unique in the target system.

- **queuesize=import-information-queue-file-size**

~ <unsigned integer> ((2-100000000)) (KB)

Specify the file size of the import information queue files specified with the `qufile001` to `qufile008` operands. The file size specified in the `queuesize` operand applies to each of the files specified with the `qufile001` to `qufile008` operands.

For details about the formula for determining the size of an import information queue file, see *4.7.7 Designing the target Datareplicator's resources*.

If you handle the import information queue files as large files, specify at least 2097152 (2 GB). To handle large files, you must already have set the OS and Datareplicator file system areas to support large files. For details, see *6.11 Handling of large files*.

Note:

If you change the size of an existing import information queue file, you must initialize the target Datareplicator. Therefore, before you change file sizes, make sure that replication has been completed.

- **reffile="import-definition-filename"**

~ <[*pathname* /]*filename* of 1-64 bytes>

Specify the name of the import definition file containing the definitions of import processing, as an absolute or relative pathname. If you specify a relative pathname, Datareplicator assumes `$HDSPATH/relative-pathname` as the absolute pathname. In this case, specify the pathname so that the final absolute pathname will not exceed 125 characters.

If the source and target tables have exactly the same format, name, and column names, you can omit the `reffile` operand. The specified filename must be unique in the target system.

- **statsfile="import-status-filename"**

~ <[*pathname* /]*filename* of 1-64 bytes>

Specify the name of the status file to be used to store information needed for error recovery processing, as an absolute or relative pathname. If you specify a relative pathname, Datareplicator assumes `$HDSPATH/relative-pathname` as the absolute pathname. In this case, specify the pathname so that the final absolute pathname will not exceed 125 characters. The specified filename must be unique in the

target system.

If you rename a status file but a file with the same name already exists, use the `hdsstart -i -f` command to execute an initial start of the target Datareplicator.

■ **statssize=import-status-file-size**

~ <unsigned integer> ((72-2000000)) (KB)

Specify the size of the import status file. For details about the size of the import status file, see 4.7.7 *Designing the target Datareplicator's resources*.

■ **unreffile1="unimported-information-filename-(primary)"unreffile2="unimported-information-filename-(secondary)"**

~ <[pathname/]filename of 1-64 bytes>

Specify the names of the unimported information files, as absolute or relative pathnames. If you specify a relative pathname, Datareplicator assumes `$HDSPPATH/relative-pathname` as the absolute pathname. In this case, specify the pathname so that the final absolute pathname will not exceed 125 characters.

There is a primary (`unreffile1`) and a secondary (`unreffile2`) unimported information file. Datareplicator uses these two files to accumulate information by swapping them. The specified filenames must be unique in the target system.

■ **unreffilesz=unimported-information-file-size**

~ <unsigned integer> ((1-32767)) <<16>> (KB)

Specify the maximum size of an unimported information file. This size applies to both `unreffile1` and `unreffile2`. For details about the size of the unimported information files, see 4.7.7 *Designing the target Datareplicator's resources*.

■ **startmode={ trn|tbl|spd }**

Specify the import method to be used when the target Datareplicator starts. The import method specified in the `startmode` operand is used only when the previous termination mode was normal termination, event termination, or immediate termination. If the target Datareplicator was terminated forcibly by the `hdsstop -t force` command or because of an import processing error, the import method used before the termination is inherited.

`trn`

Start import processing using the transaction-based import method.

`tbl`

Start import processing using a table-based import method (table-based partitioning method, key range-based partitioning method, or hash partitioning method). If `tbl` is selected, a transaction occurs for each table's import group. Therefore, if the source system updates multiple tables with a

single transaction, the target system updates the tables with separate transactions, which means that synchronization is not assured. If the tables must be synchronized, define them in the same import group.

`spd`

Start reception processing only and keep import processing inactive.

■ **restartmode=initial|continue**

Specify whether the import processing status at the time of the previous termination is to be inherited when the target Datareplicator starts (whether import processing is to be active or inactive).

If the import processing was terminated by the `hdsrfctl -m immediate` command or due to detection of an import stop event, the previous import processing status is inherited. If the import processing was terminated by the `hdsstop` command or because of an import processing error, the previous import processing status is not inherited.

`initial`

Start import processing on the basis of the `startmode` operand's specification without inheriting the import processing status in effect at the time of the previous termination.

`continue`

Start import processing by inheriting the import processing status in effect at the time of the previous termination.

■ **breaktime=hh:mm**

Specify the amount of time (hours and minutes) that is to elapse from startup of the target Datareplicator to startup of import processing (applicable when `spd` is specified in the `startmode` operand). The amount of time specified is relative to startup of the target Datareplicator. Specifying more than 23:59 or less than 00:00 will result in an operand specification error. If you omit the `breaktime` operand or specify `breaktime=00:00`, import processing will not start until it is started by entry of the `hdsrfctl` command.

hh ~ <unsigned integer> ((00-23))

mm ~ <unsigned integer> ((00-59))

hh (hours) and *mm* (minutes) must each be specified as two digits. For example, to start import processing after one hour and one minute, specify 01:01.

■ **breakmode=trn|tbl**

Specify the import method to be used when import processing is started after

startup of the target Datareplicator. This operand is applicable when the following two conditions are satisfied and you have specified the `breakmode` operand:

- `spd` is specified in the `startmode` operand
- A valid `breaktime` operand specification is in effect

`trn`

Start import processing using the transaction-based import method.

`tbl`

Start import processing using a table-based import method (table-based partitioning method, key range-based partitioning method, or hash partitioning method).

■ **eventtrn=transaction-based-import-event-code**

~ <unsigned integer> ((1-128)) <<1>>

Specify the event code to be used to start import processing with the transaction-based import method. You must establish the correspondence of the transaction-based import event code in the source system.

■ **eventtbl=table-based-import-event-code**

~ <unsigned integer> ((1-128)) <<2>>

Specify the event code to be used to start import processing with a table-based import method (table-based partitioning method, key range-based partitioning method, or hash partitioning method). You must establish the correspondence of the table-based import event code in the source system.

■ **eventretrn=transaction-based-import-restart-event-code**

~ <unsigned integer> ((1-128)) <<3>>

Specify the event code to be used to restart import processing, which is inactive, with the transaction-based import method. You need to establish the correspondence of the transaction-based import restart event codes in the source system.

■ **eventretbl=table-based-import-restart-event-code**

~ <unsigned integer> ((1-128)) <<4>>

Specify the event code to be used to restart import processing, which is inactive, with a table-based import method (table-based partitioning method, key range-based partitioning method, or hash partitioning method). You must establish the correspondence of the table-based import restart event code in the source system.

■ **eventspd=import-processing-stop-event-code**

~ <unsigned integer> ((1-128)) <<5>>

Specify the event code to be used to stop import processing. You must establish the correspondence of the import processing stop event code in the source system.

■ **eventcntreset=import-processing-count-reset-event-code**

~ <unsigned integer> ((1-128))

Specify the event code used for resetting of the import processing count. If resetting is to be synchronized between the source and target Datareplicators, this operand's value must be the same as the `eventcntreset` operand's value in the transmission environment definition.

This operand's value must be different from the value of any of the following operands (including their default values) in the import environment definition:

- `eventtrn` operand
- `eventtbl` operand
- `eventspd` operand
- `eventretrn` operand
- `eventretbl` operand

■ **defmerge=true|false**

Specify whether import processing is to be executed using the applicable source system's extraction definition as the import definition when no update information is defined in the import definition. The `defmerge` operand is applicable when an import definition is available.

`true`

Execute import processing on all tables that correspond to the source definition. If there is no import definition corresponding to an extraction definition, use the extraction definition as the import definition to continue import processing.

`false`

Execute import processing only on the tables that are defined in the import definition.

■ **cmtintvl=import-processing-commit-interval**

~ <unsigned integer> ((1-32767)) <<100>>

Specify the interval at which the target Datareplicator is to issue commit to the target HiRDB, expressed as a number of transactions at the source system. In the

following cases, the target Datareplicator always issues a commit to the target HiRDB regardless of the `cmtintvl` operand specification:

- An event is detected
- `PURGE TABLE` is executed during import processing

If you specify large values in the `cmtintvl`, `trncmtintvl`, and `tblcmtintvl` operands, the HiRDB server is adversely affected. For details, see 4.7.3(5) *Designing the COMMIT-issuance interval for import processing*.

Also, when a large value is specified in the `cmtintvl` operand, a large amount of the following HiRDB resources is used because the number of SQL statements executed by one transaction on the target Datareplicator increases:

- `pd_lck_pool_size`
- `pd_max_access_tables`
- `pd_log_sdinterval`
- `pd_sql_object_cache_size`

If an error occurs due to a shortage of locked resources, specify a smaller value in the `cmtintvl` operand.

■ **`trncmtintvl=import-processing-commit-interval-for-transaction-based-import-method`**

~ <unsigned integer> ((1-32767)) <<100>>

Specify the interval at which the target Datareplicator is to issue commit to the target HiRDB during import processing using the transaction-based import method, expressed as a number of transactions at the source system. In the following cases, the target Datareplicator always issues a commit to the target HiRDB regardless of the `trncmtintvl` operand specification:

- An event is detected
- `PURGE TABLE` is executed during import processing

When the transaction-based import method is used, the `trncmtintvl` operand takes precedence over the `cmtintvl` operand.

If you specify large values in the `cmtintvl`, `trncmtintvl`, and `tblcmtintvl` operands, the HiRDB server is adversely affected. For details, see 4.7.3(5) *Designing the COMMIT-issuance interval for import processing*.

Also, if a large value is specified in the `trncmtintvl` operand, a large amount of the following HiRDB resources is used because the number of SQL statements executed by one transaction on the target Datareplicator increases:

- `pd_lck_pool_size`

- `pd_max_access_tables`
- `pd_log_sdinterval`
- `pd_sql_object_cache_size`

If an error occurs due to a shortage of locked resources, specify a smaller value in the `trncmtintvl` operand.

■ **`tblcmtintvl=import-processing-commit-interval-for-table-based-import-method`**

~ <unsigned integer> ((1-32767)) <<100>>

Specify the interval at which the target Datareplicator is to issue commit to the target HiRDB during import processing using the table-based import method, expressed as a number of transactions at the source system. In the following cases, the target Datareplicator always issues a commit to the target HiRDB regardless of the `tblcmtintvl` operand specification:

- An event is detected
- `PURGE TABLE` is executed during import processing

When the table-based import method is used, the `tblcmtintvl` operand takes precedence over the `cmtintvl` operand.

If you specify large values in the `cmtintvl`, `trncmtintvl`, and `tblcmtintvl` operands, the HiRDB server is adversely affected. For details, see 4.7.3(5) *Designing the COMMIT-issuance interval for import processing*.

Also, if a large value is specified in the `tblcmtintvl` operand, a large amount of the following HiRDB resources is used because the number of SQL statements executed by one transaction on the target Datareplicator increases:

- `pd_lck_pool_size`
- `pd_max_access_tables`
- `pd_log_sdinterval`
- `pd_sql_object_cache_size`

If an error occurs due to a shortage of locked resources, specify a smaller value in the `tblcmtintvl` operand.

■ **`tblcheck=true|false`**

Specify whether the target system is to be checked when the target Datareplicator starts for the table subject to import processing. If an import definition is available, the target Datareplicator always checks to see if the table specified in the import definition exists at the target system, regardless of the `tblcheck` operand specification. If no import definitions are provided, the target

Datereplicator checks to see if the table specified in the extraction definition sent from the source system exists at the target system only if `true` is specified in the `tblcheck` operand.

`true`

Check to see if the table subject to import processing exists at the target system. This might result in an increase in the workload at the target HiRDB when the target Datereplicator starts.

`false`

Do not check.

■ **defshmsize=shared-memory-size-for-storing-definition-information**

~ <unsigned integer> ((1-2000000)) (KB)

If you are using UNIX Datereplicator:

Specify the size of the shared memory to be used to store the definition information analysis results. For details about the size of the shared memory, see *4.7.7 Designing the target Datereplicator's resources*.

If you are using Windows Datereplicator:

Specify the size of the memory-mapped file to be used to store the definition information analysis results. For details about the size of the memory-mapped file, see *4.7.7 Designing the target Datereplicator's resources*.

■ **ebcdic_type={ eck78|ekk78|eck83|ekk83 }**

Specify the EBCDIK/KEIS type for the data to be sent when the source system uses the EBCDIK/KEIS character code system or the source Datereplicator is to convert data to EBCDIK/KEIS. The `ebcdic_type` operand is applicable only when the transmitted data uses the EBCDIK/KEIS character code system.

`eck78`

Use EBCDIC/KEIS78.

`ekk78`

Use EBCDIC/KEIS78.

`eck83`

Use EBCDIC/KEIS83.

`ekk83`

Use EBCDIC/KEIS83.

■ **shiftspace_cnv=multi|single**

Specify the type of space character (double-byte or single-byte) to be used to replace two consecutive single-byte space characters in the double-byte mode. The `shiftspace_cnv` operand is applicable only when the transmitted data uses the EBCDIK/KEIS character code system.

`multi`

Convert two consecutive single-byte space characters to a single double-byte space character.

`single`

Convert two consecutive single-byte space characters to two consecutive single-byte space characters.

■ **`undefcode_cnv=multi|single`**

Specify the type of space character (double-byte or single-byte) to be used to replace an undefined Gaiji character.

`multi`

Convert an undefined Gaiji character to one double-byte space character.

`single`

Convert an undefined Gaiji character to two consecutive single-byte space characters.

■ **`ref_data_backspace=suppress|nosuppress`**

Specify `suppress` when all the conditions listed below are satisfied. If the defined lengths are the same at both the source and the target systems when `suppress` is not specified, an error results when an SQL statement is issued.

The following table shows the conditions under which an error results when an SQL statement is issued, depending on the character code system used in the source and target databases:

Extraction locale	Import locale			
	EBCDIK(C)/KEIS	SJIS	EUC	UTF-8
EBCDIK(C)/KEIS	--	N	E •A1 and B2 •A2 and B1	E •A1 and B3 •A2 and B3 •A3 and B3
SJIS	--	--		
EUC	--	N	--	
UTF-8	--	N	N	--

Legend:

N: No error results when an SQL statement is issued.

E: An error results when an SQL statement is issued:

A1: Data in the target database contains single-byte Kana characters.

A2: Data in the target database contains Gaiji characters.

A3: Data in the target database contains non-ASCII code.

B1: Column definition length in the target database does not exceed 1.5 times (3/2) the column definition length in the source database.

B2: Column definition length in the target database does not exceed 2 times (2/1) the column definition length in the source database.

B3: Column definition length in the target database does not exceed 3 times (3/1) the column definition length in the source database.

When `suppress` is specified, the CPU time might increase because Datareplicator deletes space characters after character code conversion. Therefore, if the source database's character code system is not EUC or UTF-8, we recommend that you specify `nosuppress`. If data consists of only space characters, Datareplicator regards the data as a single space character and executes import processing.

`suppress`

Eliminate all trailing space characters.

`nosuppress`

Do not eliminate any trailing space characters.

■ **`skip_sqlcode=SQLCODE[,SQLCODE] ...`**

Specify the SQLCODEs for database updating errors (SQL errors) that are to be skipped (ignored) during import processing so that import processing will continue.

The following two types of SQL errors can be skipped:

- SQL errors that occur during update processing (`EXECUTE`)
- SQL errors that occur during `INSERT` or `UPDATE` preprocessing (`PREPARE`)

Note that Datareplicator versions earlier than 05-00 cannot skip SQL errors that occur during SQL preprocessing (`PREPARE`).

You can specify a maximum of 32 SQLCODEs.

To specify a negative SQLCODE, use the minus sign. If a specified SQLCODE is for an SQL error that is subject to implicit rollback, Datareplicator ignores the skip specification and stops import processing.

This operand has no default value. If you omit this operand, Datareplicator will stop import processing whenever any SQL error occurs, regardless of the SQLCODE. For details about SQLCODEs, see the *HiRDB Version 9 UAP Development Guide*.

■ **skip_mvcelmwarn=true|false**

Specify the action to be taken by Datareplicator when UPDATE is executed on a table containing a repetition column during import processing and W is set in SQLWARN7 (the element number specified in the SET or DELETE clause of the UPDATE statement is not found in the row subject to update processing).

true

Skip SQLWARN7 and resume processing.

false

Handle it as an error and stop import processing.

■ **sqlerr_skip_info={ output|msgoutput|sqloutput|nooutput }**

Specify whether a warning message indicating a skip or an error is to be output and whether the unimported information associated with an SQL error is to be output according to the specification of the following operands:

- skip_sqlcode operand specifying that import of update information associated with specified SQLCODEs is to be skipped
- skip_mvcelmwarn operand specifying true (to skip warning and resume processing when W is set in SQLWARN7)

This operand is applicable only when the SQLCODEs are specified in the skip_sqlcode operand or when true is specified in the skip_mvcelmwarn operand.

output

Output both a warning message and the unimported information.

msgoutput

Output a warning message only.

sqloutput

Output the unimported information only.

nooutput

Output neither a warning message nor the unimported information.

■ **extract_init=check|nocheck**

Specify whether the reception sequence of update information is to be checked

when connection with the source system is established immediately after initialization.

check

Check that the source system is also initialized when connection is established with the source system immediately after initialization.

nocheck

Establish connection with the source system unconditionally after initialization.

■ **db_connect_retry_number=DB-connect-retries-count**

~ <0-255> <<0>> (count)

Specify the number of times connection establishment might be reattempted when the source Datareplicator's attempt to establish connection with the target database results in an error.

■ **db_connect_retry_interval=DB-connect-retry-interval**

~ <0-3600> <<30>> (seconds)

Specify in seconds the retry interval when the source Datareplicator attempts to establish connection with the target database. This operand is applicable only when 1 or a greater value is specified in the `db_connect_retry_number` operand.

■ **skip_codecvt_error=true|false**

Specify whether to stop import processing or to skip the update information containing erroneous extraction data and resume import processing when character code conversion on extraction data being processed for import results in an error.

true

In the event of an error, skip the update information containing the erroneous extraction data and resume import processing.

false

In the event of an error, stop import processing.

■ **int_trc_getl=activity-trace-collection-range**

Specify the range of activity trace information that is to be collected for each data linkage identifier at this system. This operand is applicable only when `nc` is specified in the second parameter of the `int_trc_lvl1` operand in the import system definition.

- Value for the activity trace collection range

Value	Information to be collected			
	MST (control)	RCV (reception)	RFC (import)	SQE (SQL execution)
c1	Y	Y	Y	--
c2	Y	Y	--	Y
c3	Y	Y	--	--
c4	Y	--	Y	Y
c5	Y	--	Y	--
c6	Y	--	-	Y
c7	Y	--	-	--
<i>activity-trace-collection-range</i> omitted	Y	--	--	--

Y: Collected.

--: Not collected.

■ **device01=Datareplicator-file-system-area-name[[,allocation-file-type] ...]**

~ <absolute pathname of 1-125 bytes>

device02=Datareplicator-file-system-area-name[[,allocation-file-type] ...]

~ <absolute pathname of 1-125 bytes>

:

device09=Datareplicator-file-system-area-name[[,allocation-file-type] ...]

~ <absolute pathname of 1-125 bytes>

Specify the names of the Datareplicator file system areas and the file types to be allocated in each file system area. Specify a two-digit sequence number for xx in devicexx. The following table shows the correspondence between files and the file types to which they are allocated:

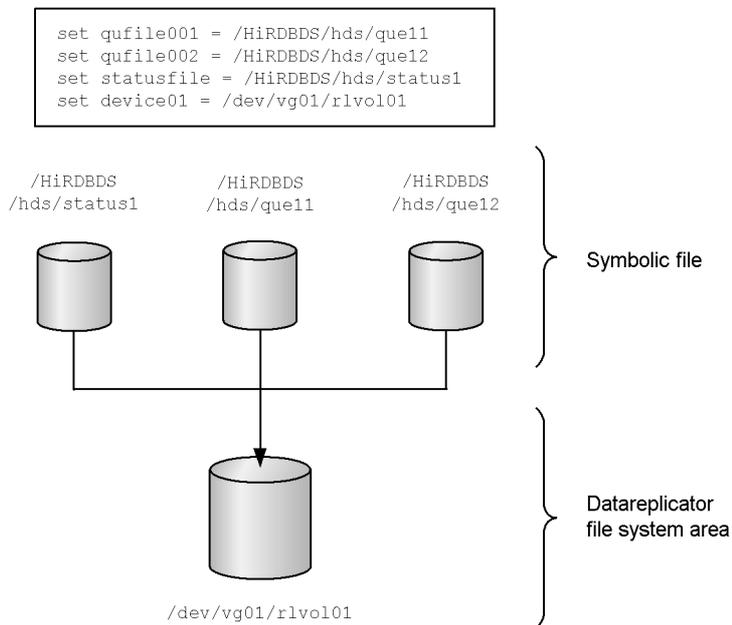
Allocation file type	File name specified in each operand
qufilexxx (xxx: integer from 001 to 008)	Link file name for import information queue file specified in set qufilexxx
statsfile	Link file name for import status file specified in set statsfile

If you do not specify any allocation file types, all files at the corresponding dsid become subject to allocation. If a file system area is specified in a devicexx

operand, Datareplicator automatically creates it and links it to the Datareplicator file system. You must ensure that the name of the file corresponding to an allocation file type has not already been allocated as a character special file.

The following is a definition example of the `devicexx` operand:

- Definition example 1: For a single server

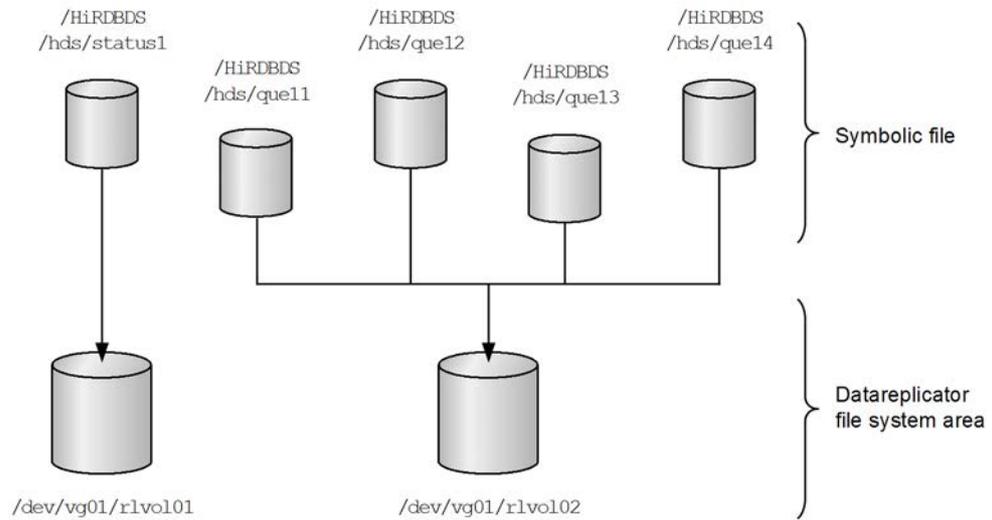


• Definition example 2: For a parallel server

```

set qfile001 = /HiRDBDS/hds/que11
set qfile002 = /HiRDBDS/hds/que12
set qfile003 = /HiRDBDS/hds/que13
set qfile004 = /HiRDBDS/hds/que14
set statusfile = /HiRDBDS/hds/status1
set device01 = /dev/vg01/rlvol01,statsfile
set device02 = /dev/vg01/rlvol02,qfile001,qfile002,qfile003,qfile004

```

*Note:*

To omit a `devicexx` operand from the definition after Datareplicator has been operated using the `devicexx` operand:

1. Stop Datareplicator.
2. Delete the `devicexx` operand from the import environment definition.
3. Use the `hdsfmkfs` command to initialize the Datareplicator file system area that was specified with the deleted `devicexx` operand.
4. Execute initial start on Datareplicator.

If you delete a `devicexx` operand from the definition, and then execute initial start on Datareplicator without following the above procedure, an invalid file error might occur during execution.

- **ujcodekind=rcv|sam**

Specify whether the data linkage identifier is to be used with the normal reception facility or to input update information. When the target Datareplicator establishes data linkage with a mainframe database that uses SAM files (PDM2 E2 or RDB1 E2), you must specify `sam` in the `ujcodekind` operand.

`rcv`

Use the data linkage identifier with Datareplicator's normal reception facility.

`sam`

Use the data linkage identifier to input update information.

- **discintvl=disconnect-issuance-interval**

~ <unsigned integer> ((0-65535)) <<180>> (seconds)

For details about this operand, see the `discintvl` operand in the import system definition. When this operand is specified, it takes precedence over the `discintvl` operand in the import system definition. If you omit this operand, the `discintvl` operand in the import system definition is effective. If you also omit the `discintvl` operand in the import system definition, Datareplicator assumes the default value for the `discintvl` operand in the import system definition.

- **ref_wait_interval=import-process's-import-information-queue-file-read-interval**

~ <unsigned integer> ((100-60000)) <<5000>> (milliseconds)

For details about this operand, see the `ref_wait_interval` operand in the import system definition. When this operand is specified, it takes precedence over the `ref_wait_interval` operand in the import system definition. If you omit this operand, the `ref_wait_interval` operand in the import system definition is effective. If you also omit the `ref_wait_interval` operand in the import system definition, Datareplicator assumes the default value for the `ref_wait_interval` operand in the import system definition.

- **commit_wait_time=COMMIT-issuance-interval**

~ <unsigned integer> ((0-300)) <<30>> (seconds)

For details about this operand, see the `commit_wait_time` operand in the import system definition. When this operand is specified, it takes precedence over the `commit_wait_time` operand in the import system definition. If you omit this operand, the `commit_wait_time` operand in the import system definition is effective. If you also omit the `commit_wait_time` operand in the import system definition, Datareplicator assumes the default value for the `commit_wait_time` operand in the import system definition.

■ **mapping_key_check= { not null unique|unique|none }**

Specify the unique condition for performing unique check on the mapping key column. This value becomes the default when the `check` clause is omitted in the import definition. Specification of this clause is effective when the `check` clause is specified in the import definition. For details about unique checking, see *check clause* in 5.10.5 *Import table definition*.

`not_null_unique`

Check the target table to confirm that the index satisfying the conditions described in Table 5-18 *Details of uniqueness checking* has been defined and its index component columns have the NOT NULL attribute.

`unique`

Check the target table to confirm that the index satisfying the conditions described in Table 5-18 *Details of uniqueness checking* has been defined. Because this option does not check the NULL value, you need to use a tool such as a user program that updates the source database to ensure the uniqueness of the data.

`none`

Do not perform a check. You need to use a tool such as a user program that updates the source database to ensure the uniqueness of the data.

Note:

If Datareplicator is not to perform checking to determine whether the target table is located at the target system, Datareplicator will not check the mapping key regardless of this operand's value. The checking to determine whether the target system contains the target table depends on the `tblcheck` operand in the import environment definition.

■ **control_trigger= execute|not execute**

Specify whether a trigger set by the user is to execute on a target table.

If a trigger is executed in the following cases, inconsistency might occur between the source and target databases:

- A BEFORE trigger uses an assignment statement of a routine control SQL.
- An AFTER trigger is defined for a target table that is to be replicated

`execute`

Execute the trigger on the target table.

`not_execute`

Do not execute a trigger on the target table.

For details about data linkage for a table for which a trigger has been set, see 3.3.6 *Data linkage for a table with a trigger set*.

■ **control_reference_trigger= execute | not_execute**

Specify whether a trigger set by HiRDB for a target table is to be executed.

`execute`

Execute a trigger set for a target table.

`not_execute`

Do not execute a trigger set for a target table.

If there is a referencing table with `CASCADE` specified as the referential constraint action, specify `not_execute`.

For details about data linkage for a table for which a trigger has been set, see 3.3.6 *Data linkage for a table with a trigger set*.

■ **check_pending= use | nouse**

Specify whether a target table with a referential constraint defined is to be placed in check pending status. This operand takes effect only when `not_execute` is specified in the `control_reference_trigger` operand.

`use`

Place such a target table in check pending status.

`nouse`

Do not place such a target table in check pending status.

■ **reflect_delay_limit_time=import-delay-period-threshold**

~ <unsigned integer> ((0-86400)) <<0>> (seconds)

Specify the threshold value for the difference (import delay period) between the time at which update information is stored in the system log file and the time at which update information is imported into the target database. If the actual import delay period exceeds the threshold value specified here, a warning message is output.

You must also treat the following operand values in the import environment definition as delay periods:

- `cmtintvl` operand
- `trncmtintvl` operand
- `tblcmtintvl` operand
- `ref_wait_interval` operand

- `commit_wait_time` operand

If a value of 0 is specified, the delay monitoring facility is disabled for import processes.

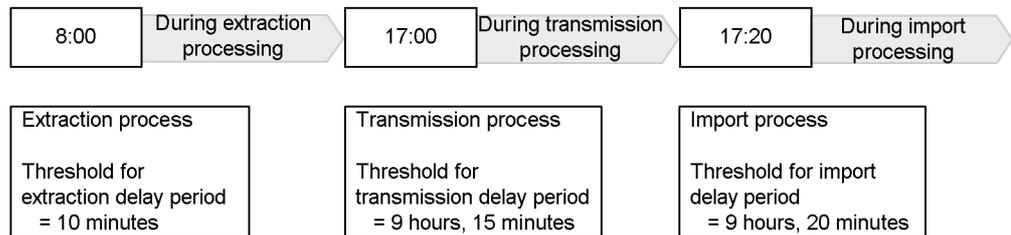
Note:

Adjust the value of this operand taking into account the amount of update information subject to replication, the transmission interval, and the import interval. A recommended guideline to start with is 20 minutes (1200 seconds).

Example

The following figure shows an example of operation when the start of transmission processing is delayed.

Figure 5-8: Example of operation where the start of transmission processing is delayed



This example executes only extraction processing from 8:00 to 17:00 and starts transmission processing after 17:00. To monitor the import delay period in such an operating environment, update data is not transmitted for up to 9 hours and is accumulated during that period at the target system. Therefore, a period of 9 hours must be added to the threshold for the import delay period.

■ `sql_lockerr_retrypnum=transaction-retry-count-in-the-event-of-a-lock-error`

~ <unsigned integer> ((0-255)) <<3>>

Specify the retry count for executing a transaction that has rolled back for one of the reasons listed below. If a value of 0 is specified, there will be no retries.

- Import processing resulted in a lock error.
- A deadlock occurred.
- A communication error occurred.

In Datareplicator, there is no retry interval because a HiRDB lock-release wait time is specified in the `pd_lck_wait_timeout` operand in the HiRDB system command definition. This operand specifies the number of retries per target

transaction; therefore, if the target transmission issues a commit, this retry count is reset.

The following describes an example in which 10 is specified in this operand:

- There will be 4 retries after a lock error occurs 4 times on a transaction.
- The next transaction does not retry for 6 times. The retry count is reset for the next transaction. Therefore, each transaction has a maximum of 10 retries for successful execution.
- The first lock error is not included in this retry count. If 10 is specified, the retry count is not exceeded until 11 lock errors occur.

The following describes when retry occurs for each target DBMS:

DBMS subject to import processing	When retry occurs	Target of retry
HiRDB	In the event of an error, <code>SQLCODE</code> is as follows: -770 (lock-release timeout error) -911 (deadlock error) -722 (communication error) -723 (communication error)	Transaction that contains the update processing resulting in the error
Oracle (Windows) SQL Server	ODBC function's return value is 40001 (deadlock error)	Transaction that contains the update processing resulting in the error
Oracle (other than Windows)	OCI function's return value is 60 (deadlock error)	Update processing resulting in the error

■ **xa_recovery_retry_count=transaction-recovery-request-retries-count**

~ <unsigned integer>((0-1024))<<180>>

Specify the number of times a transaction recovery request is to be issued in the event transaction recovery fails when a transaction recovery request has been issued to HiRDB but HiRDB has not started or there is such a large load on HiRDB's acceptance processing that HiRDB might not have been able to recover the transaction.

This operand takes effect when `fxa_sqlc` is specified in the `commitment_method` operand in the import system definition.

Specify an appropriate value using as a guideline the number of SQL processes in Datareplicator's import group (maximum number of SQL processes that can issue a recovery request concurrently to HiRDB). If 0 is specified, a transaction recovery request will not be reissued.

■ **xa_recovery_retry_interval=transaction-recovery-request-retry-interval**

~ <unsigned integer>((1-180000))<<1000>>(milliseconds)

Specify the interval (milliseconds) at which transaction recovery requests are to be issued in the event transaction recovery fails when a transaction recovery request has been issued to HiRDB but HiRDB has not started or there is such a large load on HiRDB's acceptance processing that HiRDB might not have been able to recover the transaction.

This operand takes effect when `fxa_sqlc` is specified in the `commitment_method` operand in the import system definition.

Specifying a small value in this operand reduces the time required for import processing, but CPU usage by SQL processes increases and the overall system load might become large. Specify an appropriate value taking CPU performance into account.

If the event of a large number of SQL processes in Datareplicator's import group (maximum number of SQL processes that can issue recovery requests concurrently to HiRDB), specify a larger retry interval to minimize the load created by HiRDB recovery processing.

■ **reflect_trn_max_sqlnum=maximum-update-SQL-statements-count-in-import-transaction**

~ <unsigned integer> ((1-10000000))

To import multiple source transactions as a single import transaction, specify the maximum number of update SQL statements in the import transaction. This value determines the number of source transactions to be treated as a single import transaction. If this operand is omitted, this value depends on the value of the `cmntintvl`, `trncmtintvl`, or `tblcmtintvl` operand.

Also, if a large value is specified in the `reflect_trn_max_sqlnum` operand, a large amount of the following HiRDB resources is used because the number of SQL statements executed by one transaction on the target Datareplicator increases:

- `pd_lck_pool_size`
- `pd_max_access_tables`
- `pd_log_sdinterval`
- `pd_sql_object_cache_size`

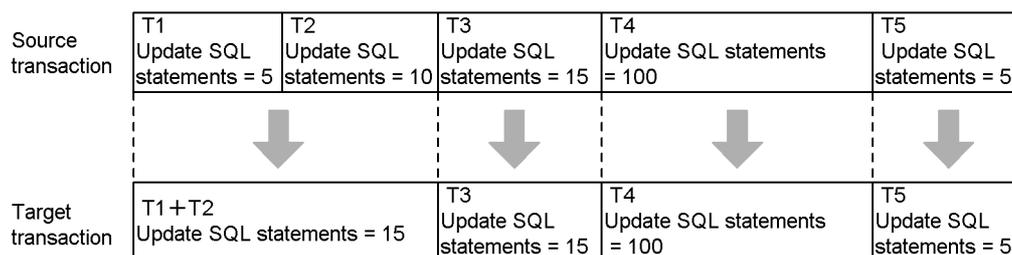
If an error occurs due to a shortage of locked resources, specify a smaller value in the `reflect_trn_max_sqlnum` operand.

The number of source transactions to be treated as a single import transaction is determined as follows:

Number of update SQL statements handled by the import transaction	Details of import transaction
<code>reflect_trn_max_sqlnum</code> operand value or less	Determined by the <code>cmtintvl</code> , <code>trncmtintvl</code> , or <code>tblcmtintvl</code> operand
Greater than the <code>reflect_trn_max_sqlnum</code> operand value	Until the total number of update SQL statements exceeds the specified value, all source transactions are handled as a single import transaction. If the specified number of update SQL statements is exceeded by a single source transaction, that transaction is handled as a single import transaction.

The following figure shows an example in which the specifications are `cmtintvl=2` and `reflect_trn_max_sqlnum=20`.

Figure 5-9: Relationship between source and target transactions



Legend:

T: Transaction

1. The number of update SQL statements for T1 is no greater than the `reflect_trn_max_sqlnum` operand value, and the `cmtintvl` operand value is 2. Therefore, whether T2 and T1 can be combined in order to be handled as a single target transaction is evaluated.

The total number of update SQL statements for T1 and T2 is no greater than the `reflect_trn_max_sqlnum` operand value. Therefore, T1 and T2 can be combined in order to be handled as a single target transaction.

2. The number of update SQL statements for T3 is no greater than the `reflect_trn_max_sqlnum` operand value, and the `cmtintvl` operand value is 2. Therefore, whether T3 and T4 can be combined in order to be handled as a single target transaction is evaluated.

The total number of update SQL statements for T3 and T4 exceeds the `reflect_trn_max_sqlnum` operand value. Therefore, T3 and T4 are handled as separate target transactions.

5. Definitions

3. The remaining T5 is handled as a single target transaction.

Note:

As a guideline for the `reflect_trn_max_sqlnum` operand's value, specify the maximum number of resources required by the target DBMS to process a single transaction.

5.10 Import definition

The import definition specifies information needed so the target Datareplicator can execute import processing. You can omit the import definition if you do not use any of the following facilities and data import is to be between tables with exactly the same format:

- Table-based import facility that executes import processing in units of groups in parallel
- Import facility that uses a UOC routine to process data before importing it
- Time-ordered information collection facility that outputs update information to a time-ordered information table

The following table shows the values that Datareplicator assumes when the import definition is omitted.

Table 5-15: Values that Datareplicator assumes when the import definition is omitted

Source database							Target database
HiRDB	XDM/SD E2 [#]	XDM/RD E2	ADM	PDM 2 E2	TMS-4V/ SP	RDB1 E2	HiRDB
Authorization identifier	Schema name	Authorization identifier	Database name	DBM name	Authorization identifier	Authorization identifier	Authorization identifier
Table identifier	Lowest-level record type	Table identifier	Segment	Dataset name	Table identifier	Table identifier	Table identifier
Column name	Component name	Column name	Field name	Field name	Column name	Column name	Column name

#

If a component name and a redefined item name in the RESTRUCT extraction redefinition statement are not unique within an extraction definition, you cannot omit the import definition because Datareplicator cannot define the import table corresponding to the redefined item name.

5.10.1 Definition rules

The following rules apply to specifying a field name, authorization identifier, and table identifier in the import definition:

- Specify the item as 1 to 30 bytes (1 to 8 bytes for an authorization identifier).
- Only the following characters can be used (for an authorization identifier, only uppercase and lowercase letters and numeric characters are permitted):
 - Uppercase letters (A-Z, #, @, \)
 - Lowercase letters (a-z)
 - Numeric characters (0-9)
 - Kana characters
 - Underscore (_)
 - Space
 - Hyphen (-)
 - Double-byte characters
- You can mix single-byte and double-byte characters.
- The first character must be a single-byte uppercase letter, single-byte lowercase letter, single-byte kana character, or a double-byte character.
- You cannot specify the double-byte space character.
- If a specification item contains a single-byte space or a single-byte hyphen, you must enclose the entire item in double quotation marks (").

5.10.2 Structure and format

(1) Structure

The following figure shows the structure of the import definition, and the table below shows the contents of the import definition and the permitted number of statements.

Figure 5-10: Structure of the import definition

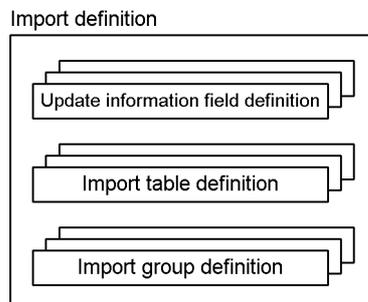


Table 5-16: Contents of the import definition and the permitted number of statements

Definition name	Definition statement	Permitted number of statements	Description
Update information field definition	<code>format statement</code>	0-4095	Establishes correspondence between extracted data and the update information fields.
Import table definition	<code>load statement</code>	0-4095	Specifies the method for importing update information and time-ordered information that is collected.
Import group definition	<code>group statement</code>	0-128	Specifies grouping for import processing when the table-based import method is used.

(2) Format

```

/* Update information field definition */
[{{ format update-information-name
  name field-name
  [{ const initial-value|by column-data-editing-UOC-function-
  identifier [nocodecnv]| [nocodecnv] ]}
  [{{ name field-name
  [{ const initial-value|by column-data-editing-UOC-function-
  identifier [nocodecnv]| [nocodecnv] ]}
  ...}}]
}} ...]

/* Import table definition */
[{{ load { field-name[{{ ,field-name }} ...]|* }
  from update-information-name
  { to[ { timestamp|sqlconvopt1|sqlconvopt2 } ]
    [ authorization-identifier.table-identifier [check{not_null_unique|
    unique|none}][with lock]|by 'uoc-name' } } } ...]

/* Import group definition */
[{{ group import-group-name
  by [authorization-identifier.table-identifier[ { { , [authorization-identifier.table-
  identifier } } ...]
  /**** Definition of key range-based partitioning method ****/
  [{{ [ in 'target-FES's-host-name' / 'target-FES's-server-identifier'
    [ / 'target-FES's-port-number' ] ]
    [ having key-range-partitioning-condition-statement[ { { ,key-range-partitioning-
    condition-statement } } ...]|other ]
  }} ...]
  /**** Definition of hash partitioning method ****/
  [{{ hash
    { in 'target-FES's-host-name' / 'target-FES's-server-identifier'

```

```

                                [ / 'target-FES's-port-number' ]
[ (RDAREA-name { { , RDAREA-name } } ... | other ) ]
    [ { { , 'target-FES's-host-name' / 'target-FES's-server-identifier'
                                [ / 'target-FES's-port-number' ] } } ... ]
[ (RDAREA-name { { , RDAREA-name } } ... | other ) ]
    | divide into SQL-process-segments-count }
    } } ... ] ]
;

```

5.10.3 Modifying defined information

To modify defined information:

1. Terminate applications at the source system.
2. Make sure that all update data has been imported into the target.
3. Terminate the source and target Datareplicators.
4. Terminate the source HiRDB normally.
5. Use a text editor to modify the extraction and import definitions.
6. Initialize the environment of the source Datareplicator.
7. Start the source HiRDB.
8. Use the `hdeprep` command to execute the extraction definition preprocess.
9. Perform an initial start of the target Datareplicator.
10. Start the source Datareplicator.
11. Restart applications at the source HiRDB.

5.10.4 Update information field definition

An update information field definition defines the name of update information to be sent from the source system and the name of each field. You can omit the update information field definition if all fields in the update information correspond exactly to the column names specified with the `load` statement in the import table definition without having to resort the fields. In this case, omit the field names and specify an asterisk in the `load` statement of the import table definition. If the asterisk (*) is specified, Datareplicator ignores the update information field definition.

You can specify only one update information field definition per unit of update information.

(1) Format

```

[ { { format update-information-name
      name field-name
      [ { const initial-value | by column-data-editing-UOC-function-
        identifier [nocodecnv] | [nocodecnv] } ] } } ]

```

```
[ { { name field-name
[ { const initial-value | by column-data-editing-UOC-function-
identifier [nocodecnv] | [nocodecnv] } ]
... } } ]
} } ... ]
```

(2) Explanation of the operands

- `format` *update-information-name*

~ <symbolic name of 1-8 characters>

Specify the update information name defined in the source system's extraction definition.

- `name` *field-name* [{ const *initial-value* | by *column-data-editing-UOC-function-identifier* [nocodecnv] | [nocodecnv] }] [{ { name *field-name* [{ const *initial-value* | by *column-data-editing-UOC-function-identifier* [nocodecnv] | [nocodecnv] }] ... } }]

Specify field names for the update information specified in the `format` statement or field names to be added. You can specify a maximum of 4000 field names.

- `name` *field-name*

Specify a field name in the update information. All field names in the update information must be specified sequentially from the beginning, regardless of whether they are to be imported. To add a field that is not in the update information, specify a desired name, and then specify `const` *initial-value* described below.

All specified field names must be unique within the update information field definition. For details about how to specify field names, see *5.10.1 Definition rules*.

- `const` *initial-value*

If the specified field name is for a new field that is to be added to the update information, specify the initial value that is to be set in that field. Note that the value specified in `const` can be imported to the target database only if the update type is `INSERT`.

The following table lists the literals that can be specified as the initial value.

Table 5-17: Literals permitted in the const clause

Literal	Column attribute	Description	Specification	Imported data
Character string literal ^{#1}	CHARACTER	Character string enclosed in single quotation marks (')	Example: 'abc'	Example: abc
Japanese-language character string literal ^{#1}	NCHAR	Japanese character string enclosed in single quotation marks (')	Example: '平和'	Example: 平和
Exact numeric literal ^{#2} or unsigned integer ^{#2}	INTEGER, SMALLINT, DECIMAL, FLOAT, SMALLFLT	Character string consisting of sign (+, -), numeric characters, and decimal point	Example: -1.23	Example: -1.23
null literal	#3	Literal indicating the null value	null	null value
Import type	CHAR	Literal indicating the update type of the import processing	reflect_kind	#4
Import date literal	DATE	Literal indicating the import date	reflect_date	Example: 19990328 ^{#5}
Import time literal	TIME	Literal indicating the import time	reflect_time	Example: 221530 ^{#6}
Import time stamp literal	TIMESTAMP	Literal indicating the import time stamp	reflect_timestamp	Example: 19990328221530 ^{#7}
Extraction date literal	DATE	Literal indicating the extraction date	extract_date	Example: 19990328 ^{#5}
Extraction time literal	TIME	Literal indicating the extraction time	extract_time	Example: 214016 ^{#6}
Extraction time stamp literal	TIMESTAMP	Literal indicating the extraction time stamp	extract_timestamp	Example: 19990328214016 ^{#7}

Note:

You can specify only the null literal in repetition columns and abstract data type columns. If you specify any other literal, a data compatibility error occurs at HiRDB during import processing.

#1: Specify a maximum of 30 bytes.

#2: Specify a maximum of 29 decimal digits.

#3: All column attributes.

#4: One of the following four types is imported, depending on the update type of the import processing:

- `upd`: Update type is updating (`UPDATE`).
- `ins`: Update type is insertion (`INSERT`).
- `del`: Update type is deletion (`DELETE`).
- `purge`: Update type is deletion of all rows (`PURGE TABLE`).

#5: The import date and extraction date are imported in the format `yyyymmdd`.

#6: The import time and extraction time are imported in the format `hhmmss`.

#7: The import time stamp and extraction time stamp are imported in the format `year-month-date-hour-minute-second` (for the second, the number of decimal places can be set at 0, 2, 4, or 6).

■ *by column-data-editing-UOC-function-identifier*

Specify the identifier of the column data editing UOC function to be used for editing when a column data editing UOC routine is used to edit the extraction data corresponding to the fields of the update information.

For the column data editing UOC function identifier, specify `typeX` (`X`: one digit in the range 1 to 8, corresponding to UOC functions `hds_ucoedit1()` to `hds_ucoedit8()`).

This operand cannot be specified for a field that corresponds to a mapping key. For details about the data types and column attributes that are not supported by column data editing UOC routines, see 8.2.5(4) *Limitations on data types*.

■ `nocodecnv`

Specify this operand to suppress character code conversion on the extraction data that corresponds to the update information field. However, this operand is not applicable to a field that corresponds to any of the following extraction column attributes:

- Field corresponding to non-character type
- Field corresponding to an abstract data type (`SGMLTEXT`, `FREWORD`, or `XML`)
- Mapping key

If the import table definition for the import information editing UOC routine (`load` statement) does not contain a field for which this option is specified, Datareplicator ignores this option and executes import processing (performs character encoding conversion processing).

5.10.5 Import table definition

An import table definition defines a table that is to be subject to import of update information. To import one update information item into multiple tables, you must create as many import table definitions as there are tables that are to be subject to the import processing. For details about the specification of field names, authorization identifiers, and table identifiers, see *5.10.1 Definition rules*.

(1) Format

```
[{ { load { field-name[ { { ,field-name } } ... ] | * }
  from update-information-name
  { to[ { timestamp|sqlconvopt1|sqlconvopt2 } ]
    [ authorization-identifier. ]table-identifier[ check {
not_null_unique|unique|none } ][ with lock ]|by 'uoc-name' }
  } } ... ]
```

(2) Explanation of the operands

- load { field-name[{ { ,field-name } } ...] | * }

Specify the fields in the update information so that they correspond to the table subject to import processing. It is essential that the field name for the mapping key corresponds to the target table's column containing the mapping key.

Specify in HiRDB's `pd_max_access_tables` operand a value that is equal to or greater than the number of specified load statements. If the `pd_max_access_tables` operand value is less than the number of specified load statements, HiRDB issues the KFP11931-E message and an error results. For details about the `pd_max_access_tables` operand, see the manual *HiRDB Version 9 System Definition*.

field-name

Specify each field name corresponding to a target table column. Specify these field names sequentially from left to right as they are to be arranged in the target table. You can specify a maximum of 4000 field names. The number of specified field names must match the number of columns in the target table.

*

If the asterisk (*) is specified, Datareplicator ignores the update information field definition for the corresponding update information (Datareplicator also ignores `const`, column data editing UOC function identifier, and `nocodecnv`).

All fields in the update information are imported into the target table in exactly the same order.

■ `from update-information-name`

~ <symbolic name of 1-8 characters>

Specify the update information name that is to be subject to import processing. This name must have been specified in the extraction definition.

■ `to[{ timestamp|sqlconvopt1|sqlconvopt2 }][authorization identifier.]table identifier`

[check { not_null_unique|unique|none }][with lock] | by 'uoc-name'

Define the authorization identifier and table identifier of the target table or a UOC name *name*'. For details about the authorization identifier and table identifier, see the manual *HiRDB Version 9 SQL Reference*.

If you specify the target table's table identifier, you can also specify the following options:

- Whether time-ordered information is to be acquired
- Whether a merge table is to be created
- Whether a uniqueness check is to be performed
- Whether the LOCK TABLE statement is to be issued to the target table

`timestamp`

Specify this operand to collect time-ordered information. When `timestamp` is specified, Datareplicator handles the information specified in the `load` clause as a time-ordered information storage table.

You cannot specify `timestamp` for a table containing a repetition column; if specified, a definition analysis error occurs and import processing with this data linkage identifier will stop.

`sqlconvopt1|sqlconvopt2`

Specify one of these values to create a merge table that enables addition and deletion as well as update processing. To specify these values, the target table must satisfy all the following conditions:

1. A unique index must be defined for some or all of the mapping key columns.
2. A unique index cannot be defined for a non-mapping key column.
3. The target table must not be a time-ordered information table.
4. The mapping key will not be updated.

The following table shows the SQL statement execution and the difference between the two values:

SQL statement	sqlconvopt1	sqlconvopt2
INSERT resulting in a duplicate key	Executes UPDATE instead.	Executes UPDATE instead.
UPDATE resulting in no corresponding row	Executes INSERT instead.	Executes INSERT instead.
DELETE	Deletes the corresponding row.	If there is a corresponding row, Datareplicator sets the NULL value in the target column that is not a mapping key column and for which the CONST clause is not specified, and then executes UPDATE. If there is no corresponding row, Datareplicator ignores the SQL statement.
PURGE	Deletes all rows.	Deletes all rows.

Notes on creating a merge table

1. To create a merge table, you must specify `sqlconvopt1` or `sqlconvopt2`. The difference between `sqlconvopt1` and `sqlconvopt2` is in the handling of DELETE data. `sqlconvopt1` handles a DELETE as row deletion, while `sqlconvopt2` handles it as null value updating. Columns that are not specified in a `format` statement's `const` clause are subject to this null value updating.
2. When you specify creation of a merge table, an SQL statement is issued twice. If there is a large volume of conversion processing, import processing might be adversely affected.
3. When a column data editing UOC routine is used, it is called twice, immediately prior to issuance of an SQL statement before and after conversion. The UOC routine is also called when DELETE is executed with `sqlconvopt2` specified; in this case, the data value is the null value.

authorization-identifier

Specify the authorization identifier for the target table. If you have specified `timestamp`, specify the authorization identifier for the time-ordered information storage table. If you omit the authorization identifier, Datareplicator assumes the authorization identifier specified in the import system definition.

table-identifier

Specify the target table's table identifier. If you have specified `timestamp`, specify the table identifier of the time-ordered information storage table.

`check {not_null_unique|unique|none}`

Specify the condition for performing a uniqueness check on the mapping key

column. The table below provides the details of uniqueness checking. When this clause is specified, Datareplicator performs the uniqueness check according to the specified value, regardless of the `mapping_key_check` operand value in the import environment definition.

When this clause is omitted, Datareplicator performs the check based on the value of the `mapping_key_check` operand.

Note that Datareplicator does not perform a uniqueness check on a table for which the `timestamp` option has been specified.

When you use the merge table creation option, we recommend that you specify `not_null_unique`.

Table 5-18: Details of uniqueness checking

Check item	Description
Index type	The index type must be one of the following: <ul style="list-style-type: none"> • Unique index • Unique cluster index • Primary index • Primary cluster index
Index component column	There must be only mapping key component columns.

`not_null_unique`

Check the target table to confirm that the index satisfying the conditions described in Table 5-18 *Details of uniqueness checking* has been defined and its index component columns have the NOT NULL attribute.

`unique`

Check the target table to confirm that the index satisfying the conditions described in Table 5-18 *Details of uniqueness checking* has been defined. Because this option does not check the NULL value, you need to use a tool such as a user program that updates the source database to ensure the uniqueness of the data.

`none`

Do not perform a uniqueness check. You need to use a tool such as a user program that updates the source database to ensure the uniqueness of the data.

The target Datareplicator's processing depends on whether the `mapping_key_check` operand or the `check` clause is specified.

The following table shows examples of the `mapping_key_check` operand (`not_null_unique` specified) and the target Datareplicator's processing.

Example	Datereplicator's processing
load * from t1 to T1 check not_null_unique	Datereplicator performs the not_null_unique check.
load * from t1 to T1 check none load * from t1 to T1	Datereplicator does not perform checking.

The following table shows examples in which the `mapping_key_check` operand is omitted (specification of the check clause takes effect) and the target Datereplicator's processing.

Example	Datereplicator's processing
load * from t1 to T1 check not_null_unique	Datereplicator performs the not_null_unique check.
load * from t1 to T1 check none	Datereplicator does not perform checking.
load * from t1 to T1	Datereplicator performs the not_null_unique check.

with lock

Specify this clause to execute import processing after issuing the `LOCK TABLE` statement for the table specified in the `load` statement. This clause is applicable only to import processing on HiRDB. If it is specified for import processing on a target other than HiRDB, this clause is ignored.

Specify the `with lock` clause in the following cases:

- Import processing is to be executed on a shared table that has been created in HiRDB's shared RDAREA (required[#])
- Overhead for locking is to be reduced during import processing (optional)

The following table describes the operation in the event that the `LOCK TABLE` statement results in an error.

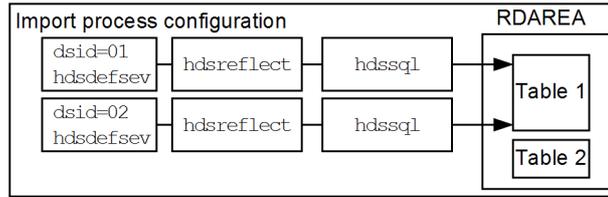
Table 5-19: Operation in the event that the LOCK TABLE statement results in an error

SQLCODE	When the skip_sqlcode operand is omitted in the import environment definition	When the skip_sqlcode operand is specified in the import environment definition
-770, -911, -722, -723	Datareplicator rolls back the executing transaction and retries the same transaction up to two times. If the second retry does not correct the error, Datareplicator outputs to the unimported information file the KFRB03034-E message and the LOCK TABLE statement resulting in the error, and then terminates import processing.	Datareplicator skips the LOCK TABLE statement and resumes import processing in the status in which no LOCK TABLE statement has been issued. The type of information to be output depends on the sqlerr_skip_info operand specified in the import environment definition (if output or msgoutput is specified, Datareplicator outputs the KFRB03043-W message).
Other	Datareplicator outputs to the unimported information file the KFRB03034-E message and the LOCK TABLE statement resulting in the error, and then terminates import processing.	

Notes:

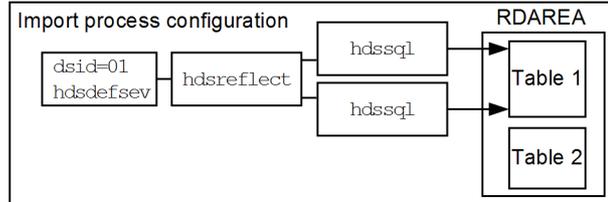
- When you specify the with lock clause, we recommend that you store a single target table per RDAREA.
- This clause is not applicable to the import process configuration shown below because deadlock might occur:

(1)



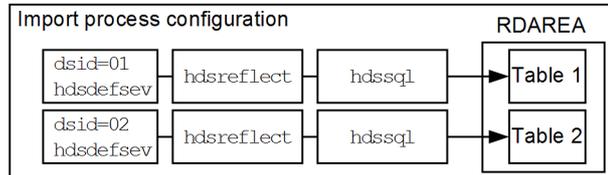
- Type of RDAREA: Shared or non-shared
- Definition: Importing from different data linkage identifiers to the same table

(2)



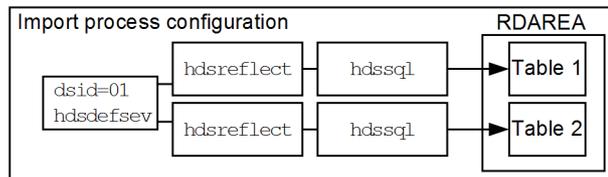
- Type of RDAREA: Shared or non-shared
- Definition: Importing using hash partitioning and key-range partitioning with the table-based import method

(3)



- Type of RDAREA: Shared
- Definition: Importing from different data linkage identifiers to a shared table in the same shared RDAREA

(4)



- Type of RDAREA: Shared
- Definition: Importing from different import groups to a shared table in the same shared RDAREA by using the table-based import method

- If the target table is a shared table, the target table and any non-target tables must be stored in different shared RDAREAs; otherwise, such a non-target table cannot be referenced or updated while an import transaction is accessing the shared table.

- `trngroup`
- `othergrp`
- `uocxxx#`

#: *xxx* is a three-digit unsigned integer.

In the case of `trngroup`, for example, you cannot specify names such as `trngroup`, `TRNGROUP`, or `TrnGroup`.

- `by`
[*authorization-identifier* .] *table-identifier* [{ { , [*authorization-identifier* .] *table-identifier* } } . . .]

Specify the target tables that are to constitute the group.

authorization-identifier

Specify the authorization identifier for a target table that is to be a member of the group. If you omit the authorization identifier, Datareplicator assumes the authorization identifier specified in the import system definition.

table-identifier

Specify the table identifier of a target table that is to be a member of the group. This table identifier must be unique in all groups. In the case of key range partitioning or hash partitioning, you can specify only one table identifier in the corresponding import group definition.

- `in` ' *target-FES's-host-name* ' / ' *target-FES's-server-identifier* ' [/ ' *target-FES's-port-number* ']

If multiple front-end servers are running using the multi-FES facility at the target HiRDB, specify the intended target front-end server's host name, server identifier, and port number. If you are using the multi-FES facility but you omit the `in` clause, the target HiRDB's system manager selects a target front-end server for the import group. If you are not using the multi-FES facility, Datareplicator assigns all import groups to the only active front-end server regardless of whether the `in` clause is specified.

To execute import processing on a single table in key range-based partitions, specify the `in` and `having` clauses as many times as there are key ranges. In this case, you can also specify the partitioning conditions in the `having` clause. You can specify a maximum of eight `in` clauses per import group definition. When you specify multiple `in` clauses in a single import group definition, you cannot omit the `having` clause. You can also specify multiple front-end servers at the same target within the same import group definition.

If the target HiRDB is one of multiple front-end servers, specify the host name of the applicable front-end server. To connect to the host for which `-p port-number`

is specified with `pdunit` in the HiRDB system definition (if the system switchover facility is used), you must specify that port number. The specification format is as follows:

host-name-of-target-FES: *port-number-of-target-FES*

target-FES's-host-name ~ <identifier of 1-32 characters>

Specify the host name specified in the target front-end server's `PDFESHOST` environment variable.

target-FES's-server-identifier ~ <identifier of 1-8 characters>

Specify the server identifier specified in the target front-end server's `PDSERVICEGRP` environment variable.

target-FES's-port-number ~ <identifier of 1-5 characters>

Specify the port number specified in the target front-end server's `PDNAMEPORT` environment variable.

For details about the above environment variables, see the *HiRDB Version 9 UAP Development Guide*.

The following shows the number of target front-end servers that can be specified in the `in` clause:

Table-based partitioning method: One target front-end server per group

Key range-based partitioning method: One target front-end server per range

Hash partitioning method: 2 to 8 target front-end servers per group

- `having`
key-range-partitioning-condition-statement [{ { *,key-range-partitioning-condition-statement* } } ...] | *other*

To execute import processing on a single table in key range-based partitions, specify condition statements for partitioning. You can specify only one `having` clause per `in` clause. If you specify multiple pairs of `in` and `having` clauses, Datareplicator checks them sequentially beginning with the first pair and executes import processing for the first condition that is satisfied.

If you specify a fixed-length character string (`CHAR`, `MCHAR`, or `NCHAR`), add trailing spaces to satisfy conditions so that the condition value length is equal to the definition length.

If none of the condition statements is true for the import data, import processing will not take place. To execute import processing even when none of the condition statements is true, specify `having other` at the end of each import group definition.

having key-range-partitioning-condition-statement

Specify a condition statement for establishing a key range partition. Specify the condition statement in the following format:

column-name=literal

Column value is equal to the specified literal.

column-name<>literal

Column value is not equal to the specified literal.

column-name>literal

Column value is greater than the specified literal.

column-name>=literal

Column value is equal to or greater than the specified literal.

column-name<literal

Column value is smaller than the specified literal.

column-name<=literal

Column value is equal to or smaller than the specified literal.

column-name IS NULL

Column value is null.

column-name IS NOT NULL

Column value is not null.

You can specify a maximum of eight condition statements per *having* clause. When you specify multiple condition statements in one *having* clause, Datareplicator joins all the specified conditions by AND. For example, if the specification is *having c1=100 c2=200*, Datareplicator executes import processing only on the data that satisfies both conditions *c1=100* and *c2=200*.

having other

Specify this operand to import data even when none of the specified key range partitioning condition statements is true. You can specify *other* only once at the end of each import group definition. Specifying *other* more than once will result in an error during analysis of the import definition. Specifying a *having* clause after *other* will also result in an error.

■ *hash*

Specify this operand to distribute the update information among the processes when multiple SQL processes are executed in parallel for import processing on a

single table. You use HiRDB's hash partitioning function when you want to distribute the update information. The hash key column is retrieved from HiRDB. If the retrieved column is not the mapping key, an error occurs during analysis of the import definition.

This hash partitioning is applicable only to a table that is hash-partitioned by HiRDB. Specifying `hash` for any other table will result in an error during analysis of the import definition. A source column and the target hash key column must satisfy the following conditions:

- The source and target columns be of the same data type.
- If the columns' data type is decimal, the source and target columns must have the same precision and scaling.
- If the columns' data type is character, the length of the source column must be equal to or less than the length of the target column.
- If the column's data type is `TIMESTAMP`, the decimal places are the same.

An error occurs during analysis of the import definition if all these conditions are not satisfied. Because Datareplicator uses HiRDB's client library when it executes hash partitioning using the `hash` option, you must add `$PDDIR/client/lib` to the `SHLIB_PATH` environment variable.

■ *RDAREA-name*

Hash partitioning involves assignment of data storage for each RDAREA; therefore, specify the RDAREA name as the assignment condition for each target FES specification. If this specification is omitted, Datareplicator determines the destination automatically.

You can omit the RDAREA name, but in such a case you must make sure that a target FES specification with an RDAREA name is not mixed with a target FES specification without an RDAREA name in the same `group` statement. If they are mixed in the same `group` statement, a definition analysis error will occur and import processing will be terminated.

A single `group` statement can contain only one RDAREA name. If multiple RDAREA names are specified in the same `group` statement, a definition analysis error will occur and import processing will be terminated.

To import data in all RDAREAs that are not specified as a condition, specify `other` as the RDAREA name. When `other` is omitted, Datareplicator assumes that `other` is specified for the last target FES specification in the `group` statement. When specifying `other`, observe the following rules:

1. Specify `other` at the end of the last `in` clause.

Correct:

```
in 'host1'/'fes1'(rdarea1,rdarea2),
    'host2'/'fes2'(rdarea3,rdarea4),
    'host3'/'fes3'(other)
```

Wrong:

```
in 'host1'/'fes1'(rdarea1,rdarea2),
    'host2'/'fes2'(other),
    'host3'/'fes3'(rdarea5,rdarea6)
```

In the example of a wrong specification, you can avoid the error by swapping lines 2 and 3.

2. other cannot be specified together with another RDAREA.

Wrong (1):

```
in 'host1'/'fes1'(rdarea1,rdarea2),
    'host2'/'fes2'(rdarea3,rdarea4),
    'host3'/'fes3'(rdarea5,other)
```

In the example of wrong specification (1), you can avoid the error by deleting other.

Wrong (2):

```
in 'host1'/'fes1'(rdarea1,rdarea2),
    'host2'/'fes2'(rdarea3,other),
    'host3'/'fes3'(rdarea5,rdarea6)
```

In the example of wrong specification (2), you can avoid the error by swapping lines 2 and 3 and deleting other.

Unlike a key range, the `hash` option cannot be used to exclude an unspecified range from import processing. The omitted RDAREAs are applied to the last FES specification. Therefore, we recommend that you define all RDAREA names in your FES specifications without omitting any of them.

- `divide into SQL-process-segments-count`
~ <unsigned integer> ((2-8))

Specify the number of SQL processes to be executed when HiRDB will execute multiple SQL processes in parallel, such as by hash partitioning, for import processing on a table at a single server. If the specified value is less than 2 or more than 8, an error occurs during analysis of the import definition.

(3) Grouping by the import group definition

When an import group definition is specified, the target Datareplicator divides the import processing into the specified groups and executes them in parallel. You can use an import group definition to create a maximum of 128 import groups. The table below explains the use of an import group definition to create import groups. Note that if you are defining a target group that employs key range-based partitioning or hash range partitioning, make sure that `PURGE TABLE` is not executed on the table.

Table 5-20: Use of an import group definition to create import groups

Classification of import processing	Creation of import groups during import processing
Importing data into the table specified in the import group definition	Datareplicator creates groups according to specifications in the import group definition.
Importing data into a table that is not specified in the import group definition	Datareplicator creates one group for all tables subject to import processing that are not specified in the import group definition.
Importing with a UOC routine	Datareplicator creates a group for each UOC routine and automatically assigns group names in the range UOC001 to UOC128.

(4) Rules for specifying key range partitioning

The following are the rules for specifying key range partitioning conditions:

- Maximum number of key range partitions (pairs of `in` and `having` clauses)

You can specify a maximum of eight key range partitions (pairs of `in` and `having` clauses) in one import group definition. Specifying more than eight key range partitions will result in an error during definition analysis.
- Maximum number of condition statements for key range partitioning

You can specify a maximum of eight condition statements separated by the single-byte comma (,) per key range partition. Specifying more than eight condition statements will result in an error during definition analysis.
- Columns specifiable in the condition statements

You can specify in the condition statements for key range partitioning only those columns defined as the mapping key for import processing. Specifying any other column in a condition statement will result in an error during definition analysis.
- Literals permitted in the columns specified in the condition statements

You can specify columns of any data types in the condition statements for key range partitioning. However, there are limitations on the permitted literals, depending on the data type. The following table shows the literals permitted in the columns specified in the condition statements for key range partitioning.

Table 5-21: Literals permitted in the columns specified in the condition statements for key range partitioning

Data type of column	Permitted literals	Example
INTEGER	Integer in the range -2147483648 to 2147483647	123456
SMALLINT	Integer in the range -32768 to 32767	12345
DECIMAL(<i>n</i>)	Integer with a length of <i>n</i> digits (<i>n</i> : 1-38)	1234567890
DECIMAL(<i>n</i> , <i>m</i>)	Fixed-point decimal number with an integer part length in the range 0 to (<i>n</i> - <i>m</i>)	123.456789
SMALLFLT	Fixed-point format and single-precision floating-point format	1.23E10
FLOAT	Fixed-point format and double-precision floating-point format	1.2345E20
CHAR	Character string consisting of 1 to 255 characters enclosed in single quotation marks (')	'ABCDEF'
VARCHAR		
NCHAR		
NVARCHAR		
MCHAR		
MVARCHAR		
DATE	'YYYY-MM-DD', which is the character string representation format at the target HiRDB: <i>YYYY</i> : 0001 to 9999 (year) <i>MM</i> : 01 to 12 (month) <i>DD</i> : 01 to 31 (date)	'2000-01-01'
TIME	'HH-MM-SS', which is the character string representation format at the target HiRDB: <i>HH</i> : 00 to 23 (hour) <i>MM</i> : 00 to 59 (minute) <i>SS</i> : 00 to 59 (second)	'12:00:00'
INTERVAL YEAR TO DAY	± <i>yyyymmdd.</i> , which is a fixed-point decimal number with precision 8 and scaling 0: <i>yyyy</i> : 0000 to 9999 (years) <i>mm</i> : 00 to 99 (months) <i>dd</i> : 00 to 99 (days)	0000010101.

Data type of column	Permitted literals	Example
INTERVAL HOUR TO SECOND	± <i>hhmmss.</i> , which is a fixed-point decimal number with precision 6 and scaling 0: <i>hh</i> : 00 to 99 (hours) <i>mm</i> : 00 to 99 (minutes) <i>ss</i> : 00 to 99 (seconds)	-010101.
TIMESTAMP	'YY-MM-DD HH-MI-SS[.NN...N]', which is the character string representation format at the target HiRDB: <i>YY</i> : 0001 to 9999 (year) <i>MM</i> : 01 to 12 (month) <i>DD</i> : 01 to 31 (date) <i>HH</i> : 00 to 23 (hour) <i>MI</i> : 00 to 59 (minute) <i>SS</i> : 00 to 59 (second) <i>NN...N</i> : 0-999999 (maximum of 6 decimal places for the second)	'2001-01-01 10:45:30.1523'

#: In the case of date data, Datareplicator only checks to determine whether the date is after the last date of the month. Datareplicator does not check for a leap year.

- Handling of trailing space characters in a fixed-length character string data type
 To specify a fixed-length character string data type in a condition statement, you must observe the following rules as appropriate to the `ref_data_backspace` operand value in the import environment definition:
 - When `suppress` is specified
 Specify the value of the fixed-length character string data type without the trailing space characters.
 - When `nosuppress` is specified
 Specify the value of the fixed-length character string data type including the trailing space characters.

(5) Notes

If you modify an import group definition while import processing is stopped for a reason other than normal termination or event termination, a loss of conformity might occur in the import processing between before and after the modification.

- Import processing method when import processing involves a UOC routine
 If import processing involves use of a UOC routine, Datareplicator executes the processing using the table-based import method regardless of the specification of the import system definition, events, or the `hdirect1` command.

5.11 Update information definition

When a PDM2 E2 (VOS3 or VOS1) source database uses SAM files, you must specify an *update information definition* for the target Datareplicator. When the source database is RDB1 E2, an update information definition is not required. An update information definition specifies information needed to extract update information from a SAM file in PDM2 E2 log format.

5.11.1 Structure and format

(1) Structure

You must specify an operating environment definition statement, extraction redefinition statements, and extraction statements, in this order. The figure below shows the structure of an update information definition. The table below shows the contents of an update information definition and the permitted number of specifications.

Figure 5-11: Structure of an update information definition

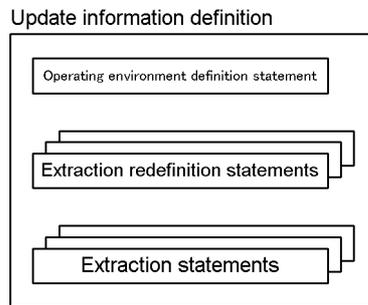


Table 5-22: Contents of an update information definition and the permitted number of specifications

Definition name	Definition statement	Permitted number of specifications	Description
Operating environment definition statement	filetype statement	1	Specifies the file type subject to processing.
Extraction redefinition statement	restruct statement	0-4096	To redefine multiple fields as one field or one field as multiple fields, this statement specifies the redefinition field name, data start position, field attribute, and length for each dataset.

Definition name	Definition statement	Permitted number of specifications	Description
Extraction statement	extract statement	1-4096	Specifies the DBM, dataset, field, update information name, and mapping key subject to extraction processing.

(2) Format

```

/* Operating environment definition statement */
filetype PDM

/* Extraction redefinition statement */
[{{ reconstruct DBM-name.dataset-name
  {{ field redefined-field-name [position data-start-position]
    attr redefined-field-attribute(length)[not null]} } ...} } ...]

/* Extraction statement */
{{ extract DBM-name.dataset-name
  ( { { field-name|redefined-field-name }
    [ { { , { field-name|redefined-field-name } } } ... ] | * } )
  to update-information-name
  key( { field-name|redefined-field-name }
    [ { { , { field-name|redefined-field-name } } } ... ] ) } } ...
;

```

5.11.2 Operating environment definition statement

The operating environment definition statement specifies information about the update information input environment.

(1) Format

```
filetype PDM
```

(2) Explanation of the operands

- **filetype PDM**

Specify the type of SAM file to be input. You can specify only one `filetype` operand per update information definition. This operand is applicable only to a SAM file in PDM2 E2 log format.

PDM

File type is a SAM file in PDM2 E2 log format.

5.11.3 Extraction redefinition statement

For a SAM file in PDM2 E2 log format, the extraction redefinition statements specify information about fields defined with PDM2 E2 to redefine multiple fields as one field or one field as multiple fields or to extract part of the data. An extraction redefinition

statement is required for each dataset. You can specify the extraction redefinition statements in any order.

If you omit an extraction redefinition statement, Datareplicator uses the field defined with PDM2 E2 in the update information definition to execute update processing.

(1) Format

```
[{{ restruct DBM-name.dataset-name
  {{ field redefined-field-name [position data-start-position]
    attr redefined-field-attribute(length) [not null]}} ...}} ...]
```

(2) Explanation of the operands

The specified DBM name, dataset name, and redefined field names must correspond to the target HiRDB's authorization identifier, table name, and column names, respectively.

- *restruct DBM-name .dataset-name*

~ <symbolic name of 1-6 characters>

In the case of the PDM2 E2 log format, specify the *DBM name* and *dataset name*. In the case of a variable dataset using the coded record facility, specify the dataset name followed by the value of the record code.

- *field redefined-field-name*

~ <symbolic name of 1-30 characters>

In the case of the PDM2 E2 log format, specify a name for a redefined field.

- *position data-start-position*

~ <<0>>

Specify the position relative to the beginning of the row data (which is 0) that is the beginning of the redefined field.

- *attr redefined-field-attribute*(*length*) [not null]

Specify a field attribute for the redefined field and the field's length. If the update data is 4040 . . . , specify *not null* to handle this data as real data, rather than as the initial value (null data). However, this operand is not applicable to packed or zoned data. The following table lists the permitted field attributes and lengths.

Table 5-23: Field attributes and lengths permitted for redefined fields

Field attribute (length)	Description	Field (bytes)
CHAR(<i>n</i>)	Character string data <i>n</i> : Number of characters (1 <= <i>n</i> <= 30000)	<i>n</i>

Field attribute (length)	Description	Field (bytes)
NCHAR(<i>n</i>)	Japanese-language character string data <i>n</i> : Number of characters ($1 \leq n \leq 15000$)	$n \times 2$
PACK(<i>m</i> [, <i>n</i>])	Signed packed decimal data <i>m</i> : Integer section. <i>n</i> : Fraction part. ($0 \leq m \leq 29, 0 \leq n \leq 29, 1 \leq m + n \leq 29$)	$(m + [n] + 1) / 2$ Value is rounded up.
PACKNS(<i>m</i> [, <i>n</i>])	Unsigned packed decimal data <i>m</i> : Integer section. <i>n</i> : Fraction part. ($0 \leq m \leq 29, 0 \leq n \leq 29, 1 \leq m + n \leq 29$)	$(m + [n] + 1) / 2$ Value is rounded up.
UNPACK(<i>m</i> [, <i>n</i>])	Signed zoned decimal data <i>m</i> : Integer section. <i>n</i> : Fraction part. ($0 \leq m \leq 29, 0 \leq n \leq 29, 1 \leq m + n \leq 29$)	$m + [n]$
UNPACKNS(<i>m</i> [, <i>n</i>])	Unsigned zoned decimal data <i>m</i> : Integer section, <i>n</i> : Fraction part ($0 \leq m \leq 29, 0 \leq n \leq 29, 1 \leq m + n \leq 29$)	$m + [n]$
COMP(<i>m</i> [, <i>n</i>])	COBOL fixed-point binary data <i>m</i> : Integer section. <i>n</i> : Fraction part. ($0 \leq m \leq 18, 0 \leq n \leq 18, 1 \leq m + n \leq 18$)	$1 \leq m + [n] \leq 4: 2$ $5 \leq m + [n] \leq 9: 4$ $10 \leq m + [n] \leq 18: 8$
BINARY(<i>m</i> [, <i>n</i>])	PL/I fixed-point binary data <i>m</i> : Precision. <i>n</i> : Scaling. ($1 \leq m \leq 31, 0 \leq n \leq 31, m \geq n$)	$1 \leq m \leq 15: 2$ $16 \leq m \leq 31: 4$
FLOAT(<i>n</i>)	Floating-point data <i>n</i> : Length in bytes ($n = 4, 8$)	<i>n</i>

5.11.4 Extraction statement

For a SAM file in PDM2 E2 log format, specify the DBM name, dataset name, and the names of the fields subject to extraction processing. If there are multiple DBM names and dataset names subject to extraction processing, specify as many extraction statements as necessary. You can specify only one extraction statement per dataset. To import a single dataset into multiple tables, specify the necessary information in the import definition.

(1) Format

```
{ { extract DBM-name.dataset-name
  ( { { field-name | redefined-field-name }
    [ { { , { field-name | redefined-field-name } } } ... ] * } )
  to update-information-name
  key( { field-name | redefined-field-name }
    [ { { , { field-name | redefined-field-name } } } ... ] ) } } ...
```

(2) Explanation of the operands

- `extract DBM-name .dataset-name`

~ <symbolic name of 1-6 characters>

In the case of the PDM2 E2 log format, specify the DBM name and dataset name. In the case of a variable dataset using the coded record facility, specify the dataset name followed by the value of the record code.

- `field-name | redefined-field-name`

`field-name` ~ <symbolic name of 1-30 characters>

Specify a field name or a redefined field name. You can specify the field names in any order, but you must not specify the same name more than once. The all-columns specification (*) is permitted only when there are no extraction redefinition statements.

- `to update-information-name`

~ <symbolic name of 1-8 characters>

Specify a name for the extracted update information. This update information name must be unique among all extraction statements contained in the update information definition.

- `key field-name | redefined-field-name`

`field-name` ~ <symbolic name of 1-30 characters>

Specify the name of a field or redefined field corresponding to the mapping key. All fields and redefined fields specified in the `key` clause must have been specified in an `extract` statement. If the all-columns specification (*) is used in the `extract` statement, each field name must be unique in the PDM2 E2 definitions.

(3) Notes

- You cannot extract only a mapping key column.
- You must specify the DBM, dataset, and field names in uppercase letters.
- Redefined field names and update information names are case-sensitive.
- To specify a Datareplicator reserved word as a DBM, dataset, field, or redefined field name, you must enclose the name in double quotation marks ("). For details about the Datareplicator reserved words, see Appendix B. *Datareplicator Reserved Words*.
- The field and redefined field names specified in the `extract` operand must be unique column names. The field and redefined field names in the `key` operand must also be unique.
- If you update the mapping key, import processing might result in an error (no data

with matching key value) or data might be imported into an unexpected row, because the target system uses the updated data as the key. For this reason, it is not advisable to update the mapping key.

- You can specify a maximum of 16 mapping key columns in the `key` operand.
- If you specify a column with either of the following attributes, you must check that the column length does not exceed 255 bytes:
 - CHAR
 - NCHAR

5.12 Duplexing definition (target)

The duplexing definition specifies the correspondence between the logical and physical file names that are used in the target system's duplexing definition.

(1) Format

```
{ { logical_file = logical-file-name
  physical_file_a = physical-file-name-1
  physical_file_b = physical-file-name-2
# comment-line
} }
:
```

The length of a single definition statement line cannot exceed 1,024 single-byte characters. To specify a comment, start the line with a hash mark (#).

(2) Explanation of the operands

`logical_file = logical-file-name`

Specify the logical file name of the file to be duplexed. The following table shows the logical file name for each file:

System	Definition file	Logical file name
Target system	Import master status file	hdsinitstate
	Import status file	Name specified in the import environment definition
	Import information queue file	Name specified in the import environment definition

If the same logical file name is defined more than once, a definition analysis error occurs. If the specified logical file name does not exist in the target environment definition file, all the definitions associated with that logical file name will be ignored.

`physical_file_a = physical-file-name-1 , physical_file_b = physical-file-name-2`

~ ((1-125)) (bytes)

Specify the absolute path names of the physical files that constitute the logical file specified in `logical_file`. If the same physical file name is defined more than once, a definition analysis error occurs.

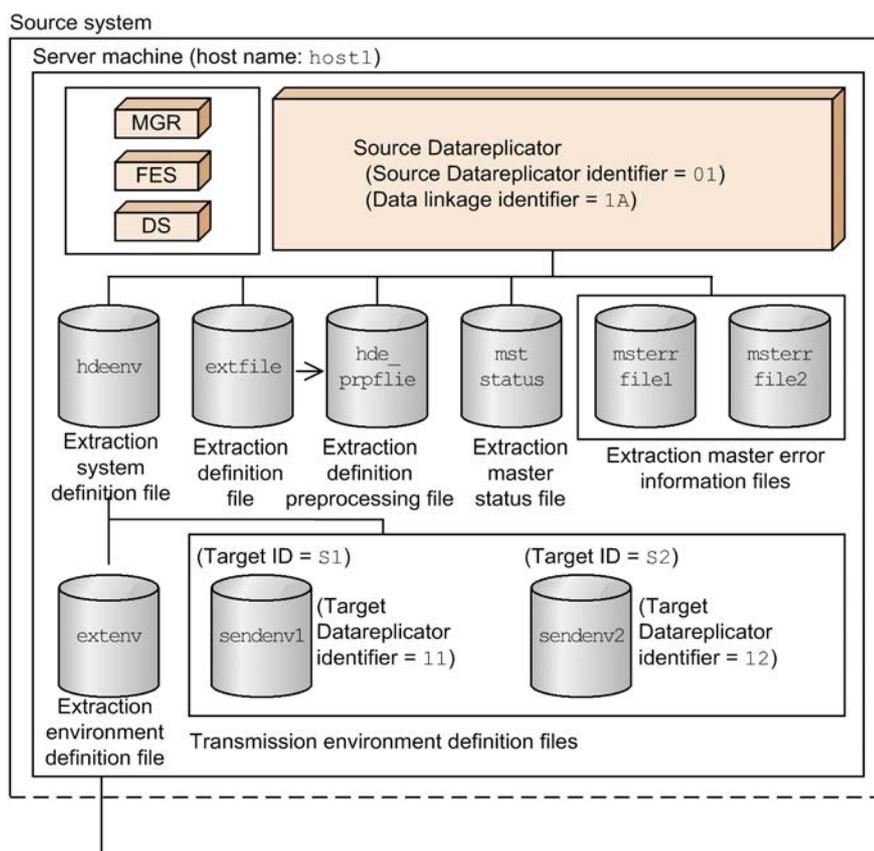
5.13 Examples of Datareplicator definitions

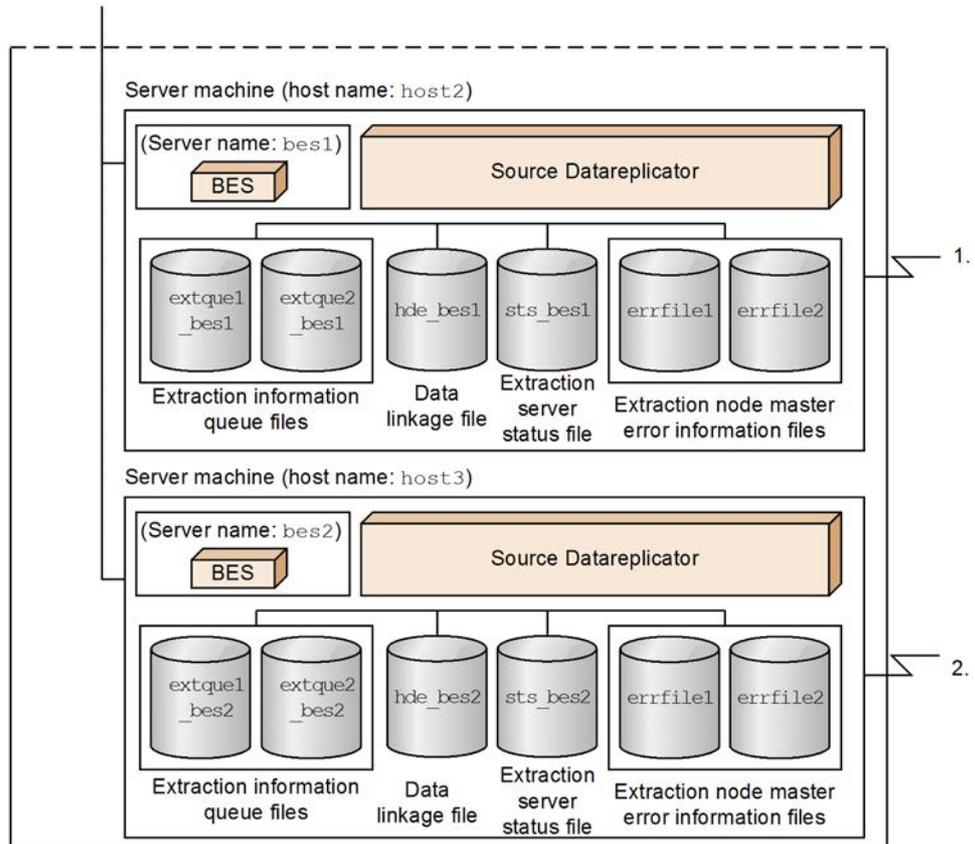
This section provides examples of Datareplicator definitions. For definition examples of a data linkage system on a mainframe system, see the *VOS3 XDM/DS* manual.

5.13.1 Examples of system configuration and file organization

The figures below show examples of system configurations and file organizations. These examples use Datareplicators at both the source system and the target system.

Figure 5-12: Example of system configuration and file organization (1)



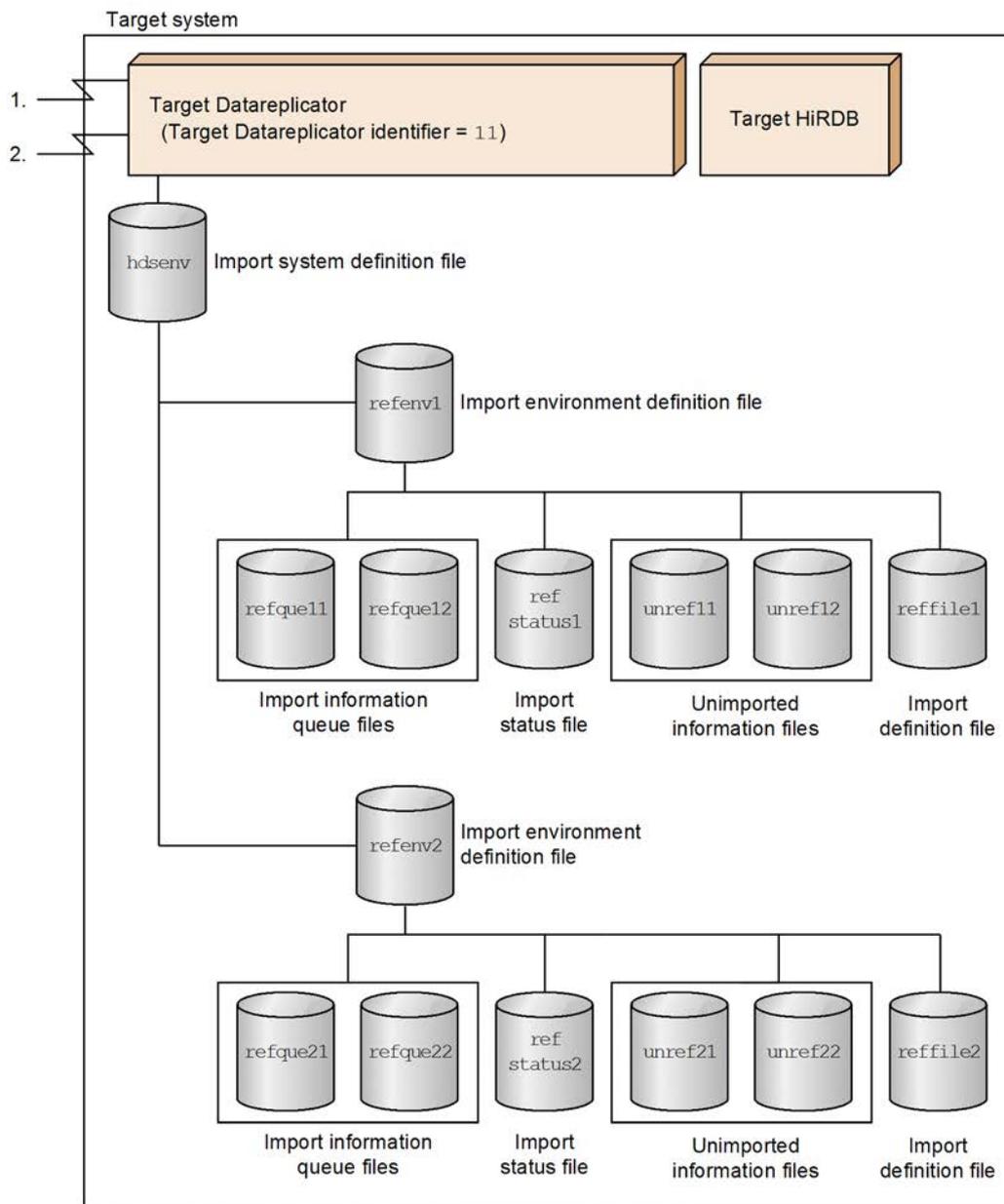


Legend:

MGR : HiRDB system manager
 FES : HiRDB front-end server

DS : HiRDB dictionary server
 BES : HiRDB back-end server

Figure 5-13: Example of system configuration and file organization (2)



5.13.2 Examples of source Datareplicator definitions

Of the source Datareplicator definitions, this section explains by way of examples the

extraction system definition, extraction environment definition, and transmission environment definition. For an example of an extraction definition, see 5.13.4 *Examples of source and target definitions*.

The source Datareplicator provides templates for source Datareplicator definitions. You can copy these templates under the `$HDEPATH/` directory and edit and use them in your user environment. To use the templates in a UNIX system with the EUC code system, you need to use OS commands for converting character codes (such as `iconv`). For details about the OS commands for converting character codes, see the applicable OS reference manual.

The templates for the source Datareplicator definitions are stored in the following directories:

- UNIX: `/opt/hirdbds/lib/sample/`
- Windows: `Datareplicator-installation-directory\sample\`

The following table shows the filenames of the templates for source Datareplicator definitions.

Table 5-24: Filenames of the templates for source Datareplicator definitions

Definition name	Filename	Specification method
Extraction system definition	<code>hdeenv</code>	<ul style="list-style-type: none"> • The templates use the JIS8/ Shift JIS code system for comments. • All lines beginning with <code>set</code> are required operands. • All lines beginning with <code>#set</code> are optional operands. • The specification range and default are indicated as <code><<specification-range; default>></code>. • If an operand has a default value, the default value is indicated as the setting; otherwise, no value is indicated. • A ruler is provided, because the maximum length of an operand is 80 characters.
Extraction environment definition	<code>extenv</code>	
Transmission environment definition	<code>sendenv</code>	
Extraction definition	<code>extfile</code>	<ul style="list-style-type: none"> • The template uses the JIS8 or Shift JIS code system for comments. • The extraction statement cannot be omitted from the extraction definition, but its specification depends on the user's definitions. Therefore, all definition examples are provided as comments. However, the semicolon (<code>;</code>) indicating the end of a definition is part of the actual coding, not a comment. • General rules are provided at the beginning.

(1) Example of an extraction system definition

The following figure shows an example of an extraction system definition.

*Figure 5-14: Example of an extraction system definition***Extraction system definition**

```

set hdeid=01           # source identifier
set extdef=extenv      # extraction environment definition
set sendid01=S1       # target identifier
set sendid02=S2
set sendef01=sendenv1 # transmission environment definition
set sendef02=sendenv2
set errfilesz=32      # error file size
set syslogout=true    # system log file
set watchintvl=30     # error monitoring interval
set mstservice=extmst # communication service name
set extinforum=1000   # maximum number of update information names
set syncterm=true     # automatic stop

```

(2) Example of an extraction environment definition

The following figure shows an example of an extraction environment definition.

*Figure 5-15: Example of an extraction environment definition***Extraction environment definition**

```

commondef
set qufile001=extque1 # common definition
set qufile002=extque2 # extraction information queue file
besdef(bes1)          # individual definition for bes1
set dsid=11           # data linkage identifier
set queuesize=4096    # queue file size
set logiosize=128     # system log i/o buffer size
set quiosize=128      # queue i/o buffer size
besdef(bes2)          # individual definition for bes2
set dsid=12           # data linkage identifier
set queuesize=2048    # queue file size
set logiosize=64      # system log i/o buffer size
set quiosize=64       # queue i/o buffer size

```

(3) Examples of transmission environment definitions**(a) Example of a transmission environment definition when the target identifier is S1**

The following figure shows an example of a transmission environment definition when the target identifier is S1.

Figure 5-16: Example of a transmission environment definition (target identifier: S1)

Transmission environment definition (target identifier: S1)

```

commondef                # common definition
set sendhdsid=11         # remote system's identifier
set hdeservice=hdeserv1 # communication service name
set hdehost=hdesht1     # target host name
set keepalive=true      # keepalive
set retrynum=5          # number of connection retries

besdef(bes1)             # individual definition for bes1
set sendintvl=20        # transmission interval
set overwrite=false     # overwrite of extraction information queue
set maxtran=200         # number of concurrent transactions
set maxtrandata=1000    # maximum number of update information items per transaction
set editbufsize=1024    # editing buffer size
set readbufnum=8        # number of queue i/o buffers

besdef(bes2)             # individual definition for bes2
set sendintvl=10        # transmission interval
set overwrite=true      # overwrite of extraction information queue
set maxtran=100         # number of concurrent transactions
set maxtrandata=1000    # maximum number of update information items per transaction
set editbufsize=512     # editing buffer size
set readbufnum=4        # number of queue i/o buffers

```

(b) Example of a transmission environment definition when the target identifier is S2

The following figure shows an example of a transmission environment definition when the target identifier is S2.

Figure 5-17: Example of a transmission environment definition (target identifier: S2)

Transmission environment definition (target identifier: S2)

```

commondef                # common definition
set sendhdsid=12         # remote system's identifier
set hdeservice=hdeserv2 # communication service name
set hdehost=hdesht2     # target host name
set keepalive=true      # keepalive
set retrynum=5          # number of connection retries

besdef(bes1)             # individual definition for bes1
set sendintvl=20        # transmission interval
set overwrite=false     # overwrite of extraction information queue
set maxtran=200         # number of concurrent transactions
set maxtrandata=1000    # maximum number of update information items per transaction
set editbufsize=1024    # editing buffer size
set readbufnum=8        # number of queue i/o buffers

besdef(bes2)             # individual definition for bes2
set sendintvl=10        # transmission interval
set overwrite=true      # overwrite of extraction information queue
set maxtran=100         # number of concurrent transactions
set maxtrandata=1000    # maximum number of update information items per transaction
set editbufsize=512     # editing buffer size
set readbufnum=4        # number of queue i/o buffers

```

5.13.3 Examples of target Datareplicator definitions

Of the target Datareplicator definitions, this section explains by way of examples the import system definition and import environment definition. For an example of an import definition, see *5.13.4 Examples of source and target definitions*.

The target Datareplicator provides templates for the target Datareplicator definitions. You can copy these templates under the `$HDSPATH/` directory and edit and use them in your user environment. To use the templates in a UNIX system with the EUC code system, you must use the OS's `stou` command (for details about the `stou` command, see the applicable OS reference manual).

The templates for the target Datareplicator definitions are stored in the following directories:

- UNIX: `/opt/hirdbds/lib/sample/`
- Windows: `Datareplicator-installation-directory\sample\`

The following table shows the filenames of the templates for target Datareplicator definitions.

Table 5-25: Filenames of the templates for target Datareplicator definitions

Definition name	Filename	Specification method
Import system definition	hdsenv	<ul style="list-style-type: none"> The templates use the JIS8 or Shift JIS code system for comments. All lines beginning with <code>set</code> are required operands. All lines beginning with <code>#set</code> are optional operands. A line beginning with <code>set</code> is a required operand. A line beginning with <code>#set</code> is an optional operand. The specification range and default are indicated as <code><<specification-range; default>></code>. If an operand has a default value, the default value is indicated as the setting; otherwise, no value is indicated. A ruler is provided, because the maximum length of an operand is 80 characters.
Import environment definition	refenv	
Import definition	reffile	<ul style="list-style-type: none"> The template uses the JIS8 or Shift JIS code system for comments. Because the entire import definition is optional, all definition examples are provided as comments. However, the semicolon (;) indicating the end of a definition is part of the actual coding, not a comment. General rules are provided at the beginning.

(1) Target system 1

(a) Example of an import system definition

The following figure shows an example of an import system definition for target system 1.

Figure 5-18: Example of an import system definition (target system 1)

Import system definition (target system 1)

```

set hdsid=11           # target identifier
set hirdbusr=ref11    # hirdb authorization identifier
set dsid001=1A        # data linkage identifier
set refenv001=refenv  # import environment definition
set hdsservice=hdsserv1 # communication service name
set keepalive=true    # keepalive
set errfilesz=32      # error file
set syslogout=true    # system log file

```

(b) Example of an import environment definition

The following figure shows an example of an import environment definition for target system 1.

Figure 5-19: Example of an import environment definition (target system 1)

Import environment definition (target system 1)

```

set qufile001=refque1 # import information queue filenames
set qufile002=refque2
set qufilesize=5000 # queue file size
set reffile=reffile # import definition filename
set statsfile=refstatus # import status filename
set unreffile1=unref1 # unimported information filenames
set unreffile2=unref2
set unreffilesz=150 # unimported information file size
set startmode=trn # start mode
set defmerge=true # use of extraction definition
set cmtintvl=20 # commit interval
set tblcheck=true # table checking
set defshmsize=500 # size of shared memory for storing definition information

```

(2) Target system 2**(a) Example of an import system definition**

The following figure shows an example of an import system definition for target system 2.

Figure 5-20: Example of an import system definition (target system 2)

Import system definition (target system 2)

```

set hdsid=12 # target identifier
set hirdbusr=ref12 # hirdb authorization identifier
set dsid001=1A # data linkage identifier
set refenv001=refenv # import environment definition
set hdsservice=hdsserv2 # communication service name
set keepalive=true # keepalive
set errfilesz=32 # error file
set syslogout=true # system log file

```

(b) Example of an import environment definition

The following figure shows an example of an import environment definition for target system 2.

Figure 5-21: Example of an import environment definition (target system2)

Import environment definition (target system 2)

```

set qufile001=refque1 # import information queue filenames
set qufile002=refque2
set qufilesize=5000 # queue file size
set reffile=reffile # import definition filename
set statsfile=refstatus # import status filename
set unreffile1=unref1 # unimported information filenames
set unreffile2=unref2
set unreffilesz=150 # unimported information file size
set startmode=trn # start mode
set defmerge=true # use of extraction definition
set cmtintvl=20 # commit interval
set tblcheck=true # table checking
set defshmsize=500 # size of shared memory for storing definition information

```

5.13.4 Examples of source and target definitions

This section explains by way of examples the following source and target definitions:

- For achieving data linkage by selecting and resorting the source table's columns at the source
- For achieving data linkage by selecting and resorting the source table's columns at the target
- For achieving data linkage by adding a column (fixed value) to the source table at the target
- For achieving data linkage by dividing one table into two tables at the target
- For achieving data linkage by dividing one table into two tables at the source
- For achieving data linkage by combining two tables into one table at the target

These examples use the source table's form number (FNO) column as the mapping key.

(1) *Selecting and resorting columns at the source*

The following figure shows an example of source and target definitions when data linkage is used for selecting and resorting the source table's columns at the source.

Figure 5-22: Example of source and target definitions: Selecting and resorting columns at the source

Source table: N1.ORDER

FNO	CCODE	PCODE	OQTY
Form No.	Customer code	Product code	Order quantity
026551	TT002	101M	10
026552	TT002	591M	25
026553	TH001	353M	8

⋮

Extraction definition

```
extract
N1.ORDER(PCODE,OQTY,FNO) /* select and resort */
to UPDATE_ORDER /* update information */
key(FNO) /* mapping key */

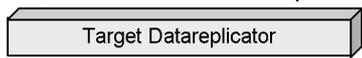
send s1 /* destination */
from UPDATE_ORDER; /* update information */
```

Update information:

UPDATE_ORDER

PCODE	OQTY	FNO
-------	------	-----

Target identifier: s1



Update information:

UPDATE_ORDER

PCODE	OQTY	FNO
-------	------	-----

Target table: N1.ORDER1

Product code	Order quantity	Form No.
101M	10	026551
591M	25	026552
353M	8	026553

⋮

Import definition

```
load
from UPDATE_ORDER /* update information */
to N1.ORDER1; /* target table */
```

(2) Selecting and resorting columns at the target

The following figure shows an example of source and target definitions when data linkage is used for selecting and resorting the source table's columns at the target.

Figure 5-23: Example of source and target definitions: Selecting and resorting columns at the target

Source table: N1.ORDER

FNO	CCODE	PCODE	OQTY
Form No.	Customer code	Product code	Order quantity
026551	TT002	101M	10
026552	TT002	591M	25
026553	TH001	353M	8

⋮

Extraction definition

```

extract
  N1.ORDER(*)           /* extract all */
  to UPDATE_ORDER      /* update information */
  key(FNO)              /* mapping key */

  send S1               /* destination */
  from UPDATE_ORDER;    /* update information */
  
```

Update information:

UPDATE_ORDER

FNO	CCODE	PCODE	OQTY
-----	-------	-------	------

Target identifier: S1



Update information:

UPDATE_ORDER

PCODE	OQTY	FNO
-------	------	-----

Target table: N1.ORDER1

Product code	Order quantity	Form No.
101M	10	026551
591M	25	026552
353M	8	026553

⋮

Import definition

```

format UPDATE_ORDER /* update order information */
name FNO             /* form field */
name CCODE           /*customer field */
name PCODE           /* product field */
name OQTY            /* ordered quantity field */

load  PCODE,OQTY,FNO /* select and resort */
from  UPDATE_ORDER
to    N1.ORDER1;    /* target table */
  
```

(3) Adding a column (fixed value) at the target

The following figure shows an example of source and target definitions when data linkage is used for adding a column (fixed value) to the source table at the target.

Figure 5-24: Example of source and target definitions: Adding a column (fixed value) at the target

Source table: N1.ORDER

FNO	CCODE	PCODE	OQTY
Form No.	Customer code	Product code	Order quantity
026551	TT002	101M	10
026552	TT002	591M	25
026553	TH001	353M	8

⋮

Extraction definition

```

extract
N1.ORDER(*)          /* extract all          */
to UPDATE_ORDER     /* update information */
key (FNO)            /* mapping key       */

send S1              /* target identifier  */
from UPDATE_ORDER ; /* update information */

```



Update information: UPDATE_ORDER

FNO	CCODE	PCODE	OQTY
-----	-------	-------	------



Target identifier: S1



Update information: UPDATE_ORDER

FNO	CCODE	PCODE	OQTY	ODATE
-----	-------	-------	------	-------



Target table: N1.ORDER2

Form No.	Customer code	Product code	Order quantity	Order date
026551	TT002	101M	10	051015
026552	TT002	591M	25	051015
026553	TH001	353M	8	051015

⋮

Import definition

```

format UPDATE_ORDER          /* update order information */
name FNO                     /* form field */
name CCODE                   /* customer field */
name PCODE                   /* product field */
name OQTY                    /* ordered quantity field */
name ODATE const '051015'   /* order date field added */

load FNO,CCODE,PCODE,OQTY,ODATE /* no resorting */
from UPDATE_ORDER
to N1.ORDER2 ;

```

(4) Dividing one table into two tables at the target

The following figure shows an example of source and target definitions when data linkage is used for dividing one source table into two tables at the target.

Figure 5-25: Example of source and target definitions: Dividing one table into two tables at the target

Source table: N1.ORDER

FNO	CCODE	PCODE	OQTY
Form No.	Customer code	Product code	Order quantity
026551	TT002	101M	10
026552	TT002	591M	25
026553	TH001	353M	8

⋮

Extraction definition

```

extract
N1.ORDER(*)          /* extract all */
to UPDATE_ORDER     /* update information */
key(FNO)            /* mapping key */

send S1             /* destination */
from UPDATE_ORDER; /* update information */
    
```

Update information:

UPDATE_ORDER

FNO	CCODE	PCODE	OQTY
-----	-------	-------	------

Target identifier: S1

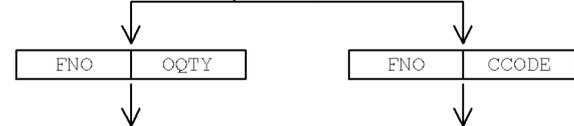


Update information:

UPDATE_ORDER

FNO	CCODE	PCODE	OQTY
-----	-------	-------	------

Import to two tables



Target table: N1.QUANTITY

Form No.	Order quantity
026551	10
026552	25
026553	8

⋮

Target table: N1.CUSTOMER

Form No.	Customer code
026551	TT002
026552	TT002
026553	TH001

⋮

Import definition

```

format UPDATE_ORDER
name FNO
name CCODE
name PCODE
name OQTY
/* import to QUANTITY */
load FNO,OQTY
from UPDATE_ORDER
to N1.QUANTITY
/* import to CUSTOMER */
load FNO,CCODE
from UPDATE_ORDER
to N1.CUSTOMER

group REFGROUP
by N1.QUANTITY,N1.CUSTOMER;
    
```

(5) Dividing one table into two tables at the source

The following figure shows an example of source and target definitions when data linkage is used for dividing one source table into two tables at the source.

Figure 5-26: Example of source and target definitions: Dividing one table into two tables at the source

Source table: N1.ORDER

FNO	CCODE	PCODE	OQTY
Form No.	Customer code	Product code	Order quantity
026551	TT002	101M	10
026552	TT002	591M	25
026553	TH001	353M	8

⋮

Divide into two and extract

Update information:

UPDATE_ORDER1

FNO	OQTY
-----	------

Target identifier: S1



Update information:

UPDATE_ORDER1

FNO	OQTY
-----	------

Target table: N1.QUANTITY

Form No.	Order quantity
026551	10
026552	25
026553	8

⋮

Update information:

UPDATE_ORDER2

FNO	CCODE
-----	-------

Target identifier: S2



Update information:

UPDATE_ORDER2

FNO	CCODE
-----	-------

Target table: N1.CUSTOMER

Form No.	Customer code
026551	TT002
026552	TT002
026553	TH001

⋮

Extraction definition

```

/* extract to UPDATE_ORDER1 */
extract
  N1.ORDER(FNO,OQTY)
  to UPDATE_ORDER1
  key (FNO)
/* extract to UPDATE_ORDER2 */
extract
  N1.ORDER(FNO,CCODE)
  to UPDATE_ORDER2
  key (FNO)
/* send to S1 */
send S1
  from UPDATE_ORDER1
/* send to S2 */
send S2
  from UPDATE_ORDER2;

```

Import definition (S1)

```

load *
  from UPDATE-ORDER1
  to N1.QUANTITY;

```

Import definition (S2)

```

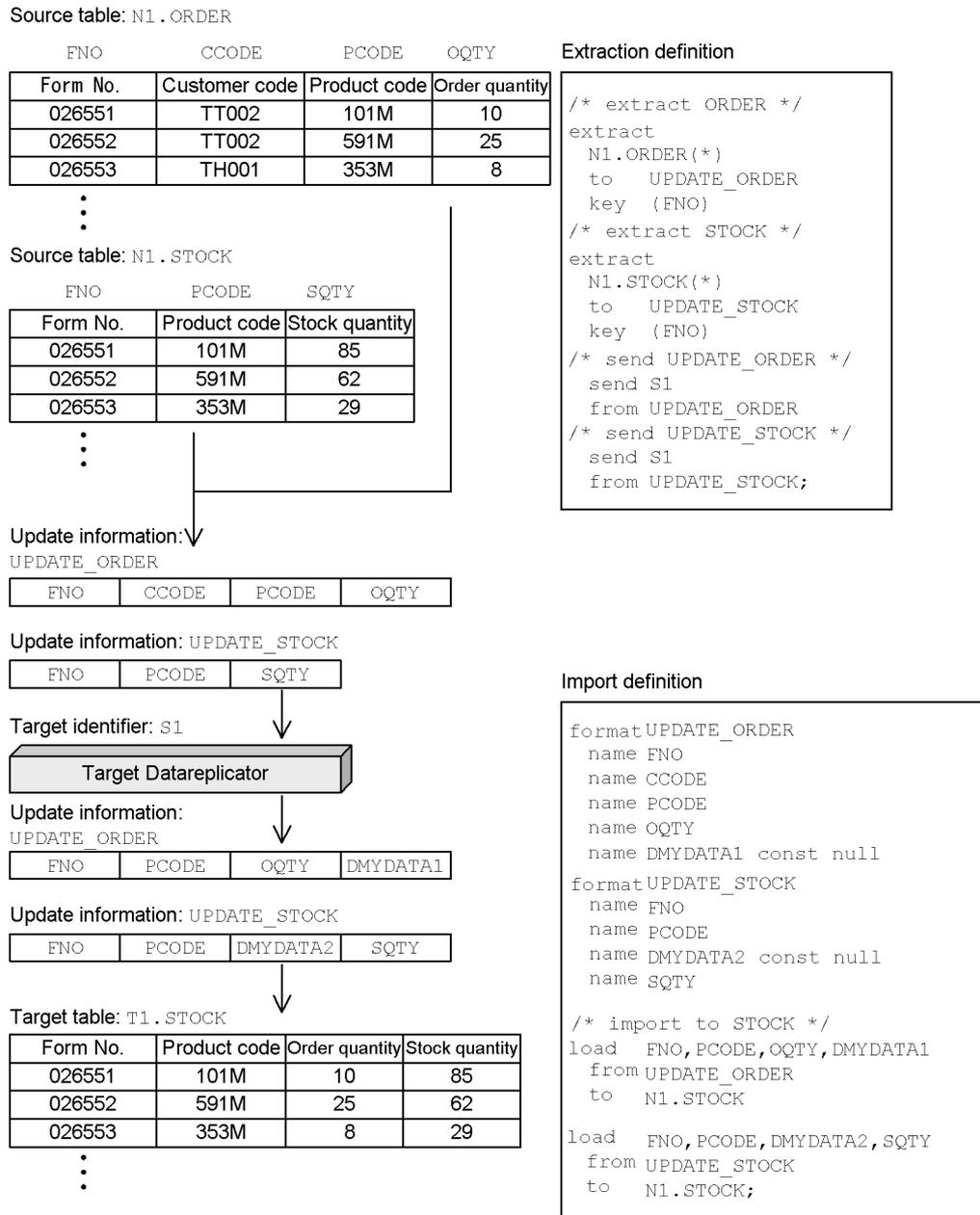
load *
  from UPDATE-ORDER2
  to N1.CUSTOMER;

```

(6) Importing two tables into one table at the target

The following figure shows an example of source and target definitions when data linkage is used for combining two source tables into one table at the target.

Figure 5-27: Example of source and target definitions: Combining two tables into one table at the target



5.13.5 Examples of import group definitions when the multi-FES facility is used

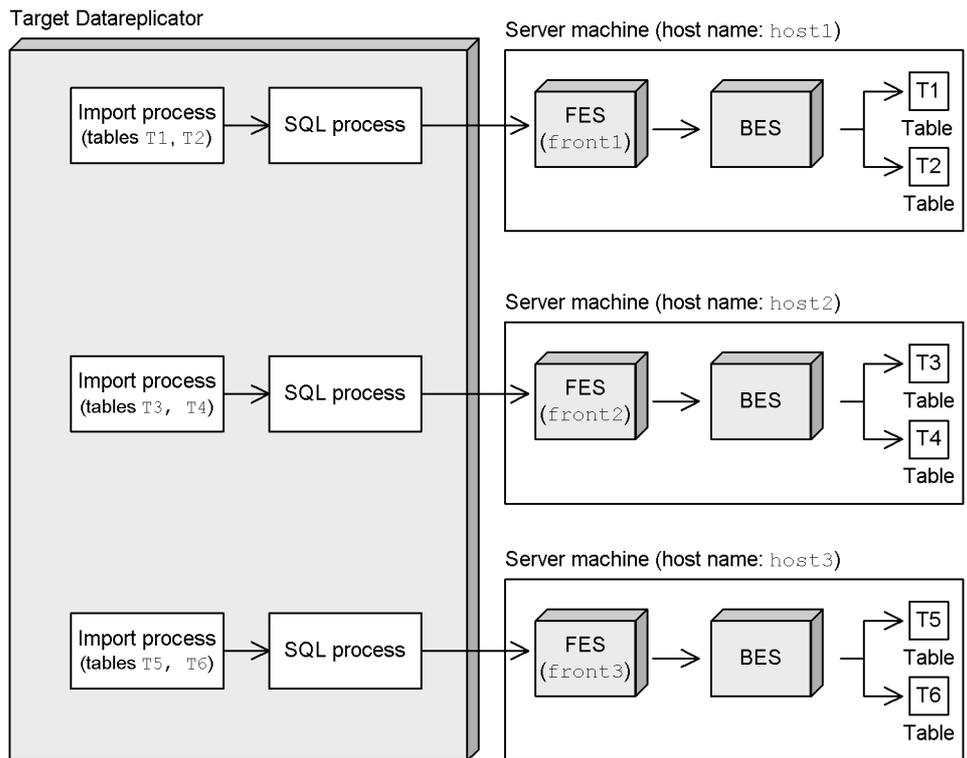
This section explains the import group definitions when the multi-FES facility is used with the following partitioning methods:

- Multi-FES facility used with the table-based partitioning method
- Multi-FES facility used with the key range-based partitioning method
- Multi-FES facility used with the hash partitioning method

(1) Multi-FES facility used with the table-based partitioning method

The following figure shows an example of an import group definition when the multi-FES facility is used with the table-based partitioning method.

Figure 5-28: Example of an import group definition: Table-based partitioning method



Definition of import groups

```

group GRP01 by T1, T2
in 'host1'/'front1'

group GRP02 by T3, T4
in 'host2'/'front2'

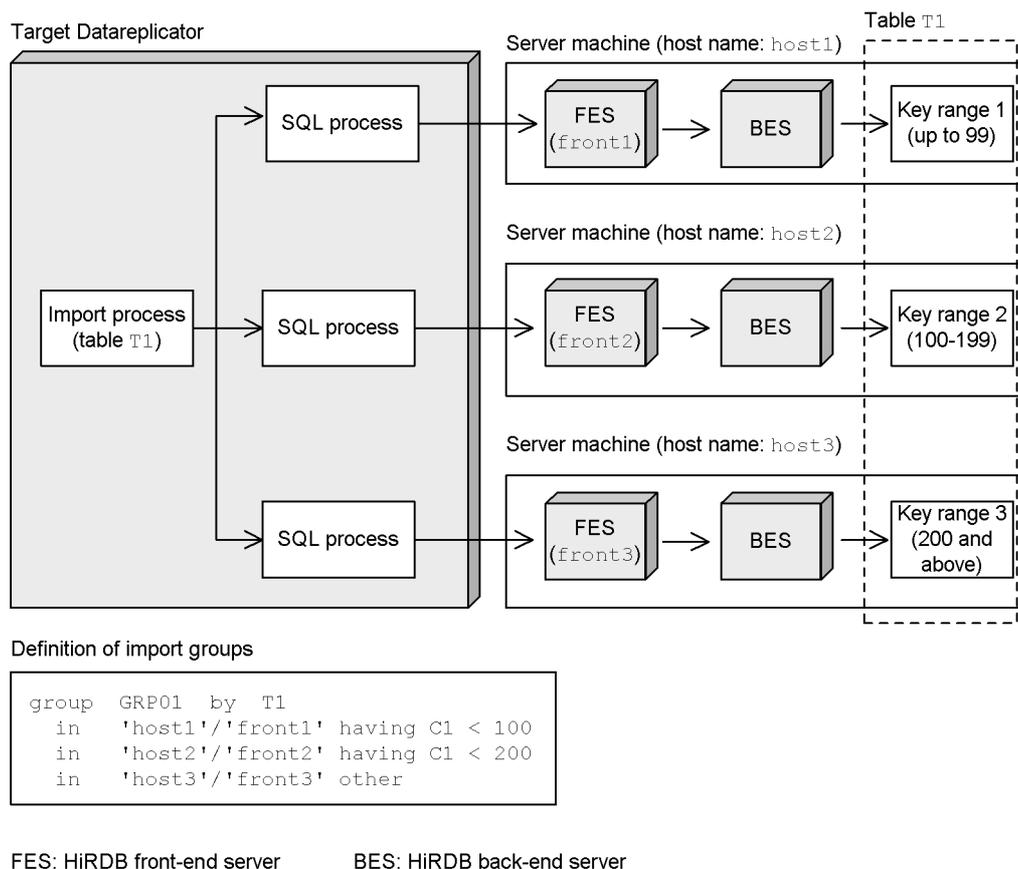
group GRP03 by T5, T6
in 'host3'/'front3'
    
```

FES: HiRDB front-end server BES: HiRDB back-end server

(2) Multi-FES facility used with the key range-based partitioning method

The following figure shows an example of an import group definition when the multi-FES facility is used with the key range-based partitioning method.

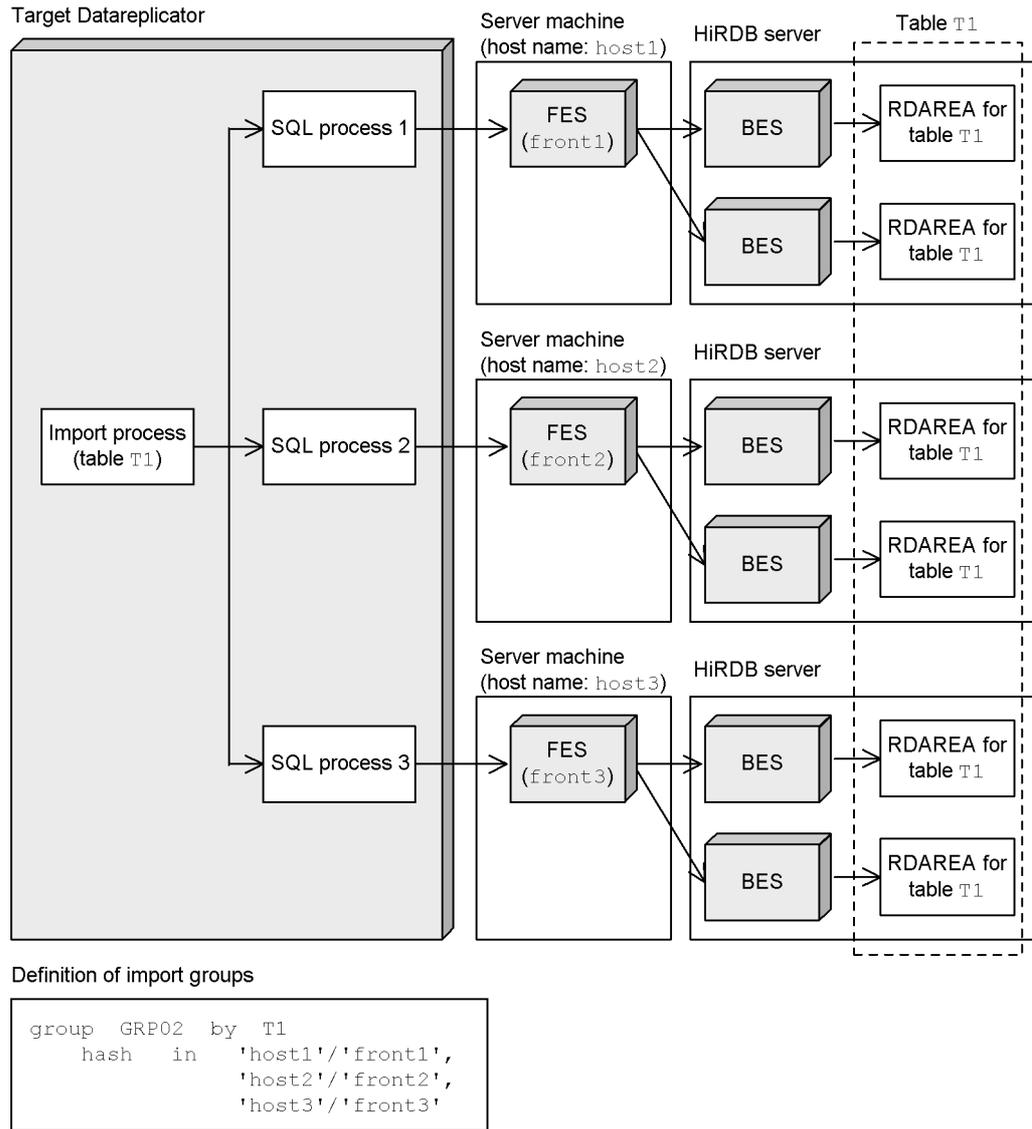
Figure 5-29: Example of an import group definition: Key range-based partitioning method



(3) Multi-FES facility used with the hash partitioning method

The following figure shows an example of an import group definition when the multi-FES facility is used with the hash partitioning method.

Figure 5-30: Example of an import group definition: Hash partitioning method



FES: HiRDB front-end server BES: HiRDB back-end server

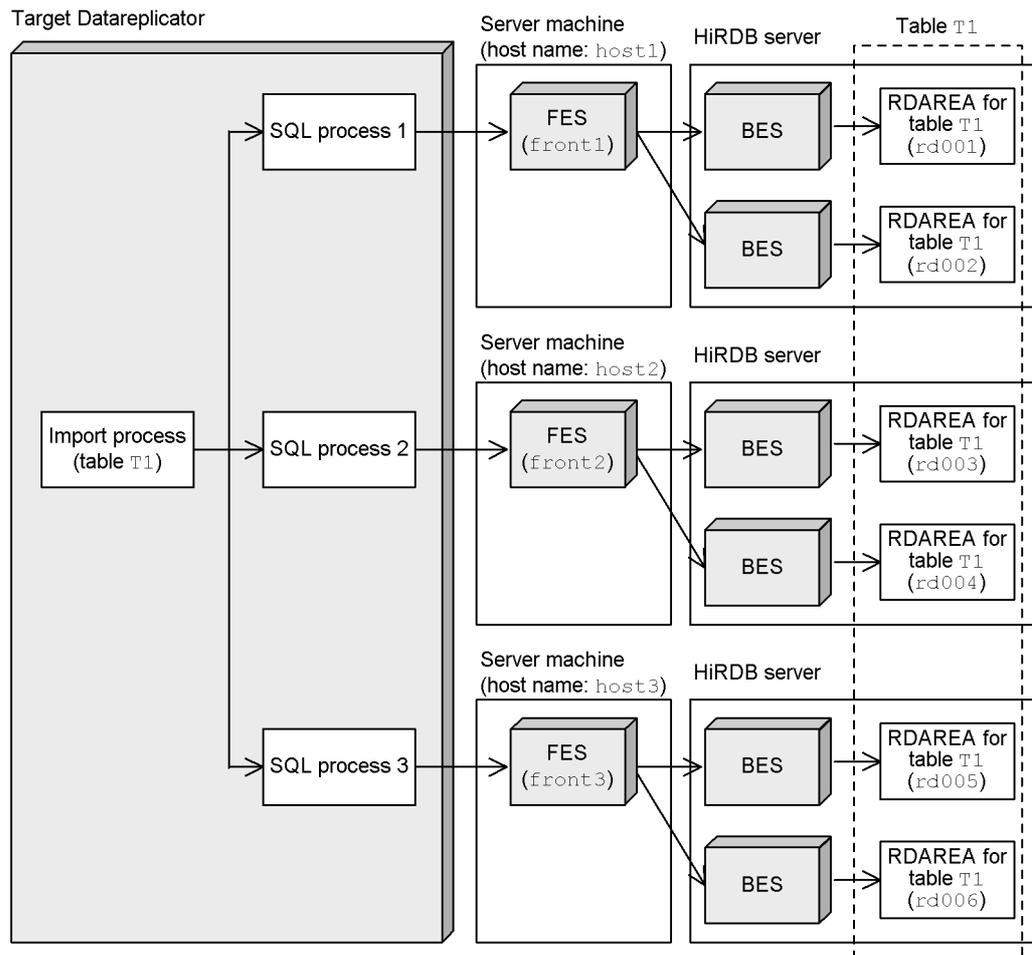
(4) Front-end servers specified with the hash partitioning method

This example specifies the front-end servers as Datareplicator's target group for a parallel server's table where data is stored in individual servers by the hash partitioning

method. By specifying the front-end servers, you can concurrently execute as many import processes as there are hash partitions; you can also reduce the communications overhead between the front-end and back-end servers, thereby reducing the overhead of the front-end servers. As a result, data linkage can be achieved more efficiently.

The following figure shows an example of an import group definition (when the front-end servers are specified with the hash partitioning method).

Figure 5-31: Example of an import group definition (when the front-end servers are specified with the hash partitioning method)



Definition of import groups

```
group GRP01 by T1
  hash in 'host1'/'front1'(rd001, rd002),
         'host2'/'front2'(rd003, rd004),
         'host3'/'front3'(other)
```

FES: HiRDB front-end server

BES: HiRDB back-end server

Chapter

6. Operation

This chapter explains the procedures for operating a data linkage system, Datareplicator startup and termination, and Datareplicator operation.

- 6.1 Overview of data linkage systems
- 6.2 Initialization procedure at the environment configuration stage
- 6.3 Startup and termination of the source Datareplicator
- 6.4 Operation of the source Datareplicator
- 6.5 Handling of the source HiRDB
- 6.6 Startup and termination of the target Datareplicator
- 6.7 Operation of the target Datareplicator
- 6.8 Changing the configuration of HiRDB and Datareplicator
- 6.9 Using the system switchover facility
- 6.10 Using the file duplexing function
- 6.11 Handling of large files
- 6.12 Tuning
- 6.13 Notes about operation

6.1 Overview of data linkage systems

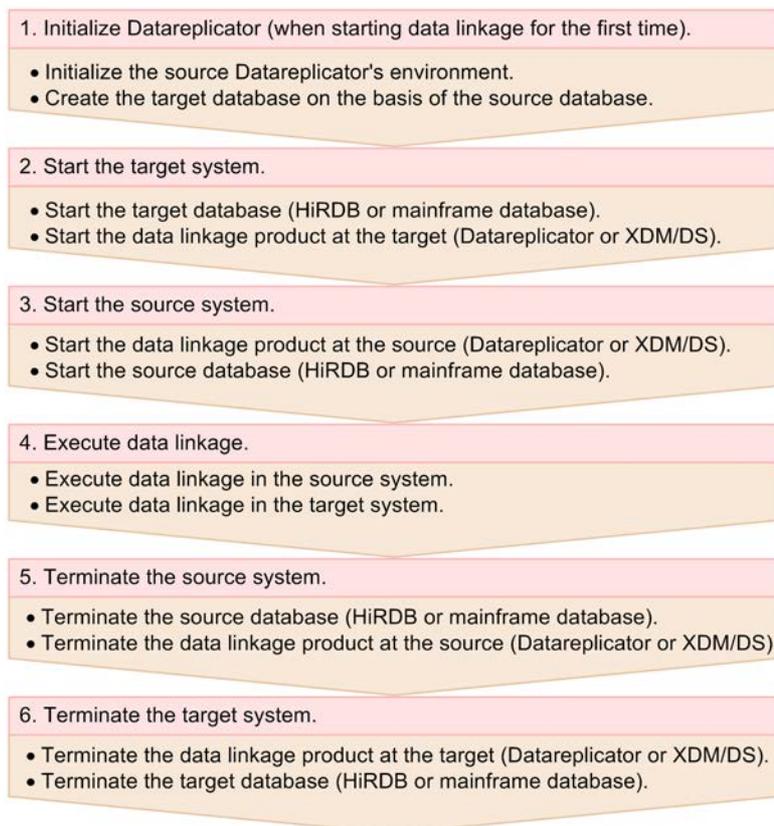
This section provides an overview of handling a data linkage system. The following table shows the combinations of data linkage systems that use Datareplicator:

Data linkage format	See
<ul style="list-style-type: none">• Data linkage from one HiRDB to another• Data linkage from a mainframe database (XDM/SD E2, XDM/RD E2, ADM, PDM2 E2, or TMS-4V/SP) to a HiRDB• Data linkage from a HiRDB to a mainframe database (XDM/RD E2)	6.1.1
<ul style="list-style-type: none">• Data linkage from a mainframe database to a HiRDB using SAM files	6.1.2

6.1.1 Normal data linkage system operating procedure

The following figure shows the normal data linkage system operating procedure.

Figure 6-1: Normal data linkage system operating procedure



(1) Initialize the Datareplicators (when starting data linkage for the first time)

You must initialize the Datareplicators when you start data linkage for the first time after you have configured the environment. For details, see *6.2 Initialization procedure at the environment configuration stage*.

(2) Start the target system

To start the target system:

1. Start the target database (HiRDB or XDM E2).
2. Start the data linkage product (Datareplicator or XDM/DS) at the target system.

If the target system is not active during data transmission, the source system checks the target system for its startup status at each transmission interval and starts transmission once it detects that the target system has started. If the source system cannot send data, its resources (such as extraction information queue files and system log file) might become full. For this reason, it is advisable to start the source system after the target

systems have completed their startup procedure.

(3) Start the source system

To start the source system:

1. Start the data linkage product (Datareplicator or XDM/DS) at the source system.
2. Start the source database (HiRDB or XDM E2).

You can start the source Datareplicator whether the source HiRDB is active. However, if the source Datareplicator is not active while the source HiRDB is running, the system log file might become full, resulting in forced termination of the source database or a data linkage error.

(4) Execute data linkage processing

When the target and source systems start normally, data linkage begins in accordance with the system definitions. For details about how to handle data linkage processing at a Datareplicator source system, see *6.4.1 Handling of extraction processing*. For details about how to handle data linkage processing at a target system, see *6.7.1 Handling of import processing*. For details about how to handle data linkage processing at XDM/DS source and target systems, see the manual *VOS3 XDM Data Linkage Facility XDM/DS Operation*.

(5) Terminate the source system

To terminate a data linkage system application, first terminate the source system.

To terminate the source system:

1. Terminate the source database (HiRDB or XDM E2).
2. Terminate the data linkage product (Datareplicator or XDM/DS) at the source system.

For details about HiRDB termination, see the *HiRDB Version 9 System Operation Guide*. For details about source Datareplicator termination, see *6.3 Startup and termination of the source Datareplicator*. For details about how to terminate XDM/DS, see the manual *VOS3 XDM Data Linkage Facility XDM/DS Operation*.

(6) Terminate the target system

Terminate the target system.

To terminate a target system:

1. Terminate the data linkage product (Datareplicator or XDM/DS) at the target system.
2. Terminate the target database (HiRDB or XDM E2).

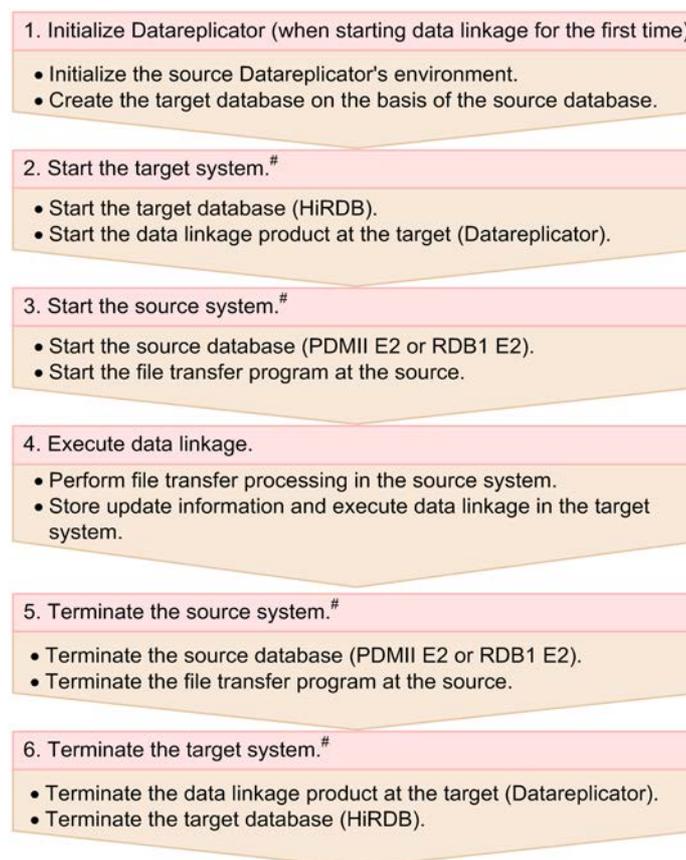
For details about target Datareplicator termination, see *6.6 Startup and termination of the target Datareplicator*. For details about HiRDB termination, see the *HiRDB*

Version 9 System Operation Guide. For details about how to terminate XDM/DS, see the manual *VOS3 XDM Data Linkage Facility XDM/DS Operation*.

6.1.2 Data linkage from a mainframe database to a HiRDB using SAM files

The following figure shows the procedure for handling data linkage from a mainframe database (PDMII E2 or RDB1 E2) to a HiRDB using SAM files.

Figure 6-2: Procedure for handling data linkage from a mainframe database to a HiRDB using SAM files



#

The order of steps 2 and 3 and the order of steps 5 and 6 can be reversed.

(1) Initialize the Datareplicators (when starting data linkage for the first time)

You must initialize the Datareplicators when you start data linkage for the first time

after you have configured the environment. For details, see *6.2 Initialization procedure at the environment configuration stage*.

(2) Start the target system

To start the target system:

1. Start the target database (HiRDB).
2. Start the data linkage product (Datareplicator) at the target system.

(3) Start the source system

To start the source system:

1. Start the source database (PDM2 E2 or RDB1 E2).
2. Start the file transfer program.

The program startup procedure at the source system depends on the product specifications.

(4) Execute data linkage processing

The source system uses a file transfer program to send a SAM file that contains update information. The SAM file is then stored in the target Datareplicator directory specified by the source system's file transfer program.

The target Datareplicator operator uses the `hdssamqin` command to store the update information from the SAM file into the import information queue file. When all update information has been stored in the import information queue file, the target system executes import processing.

(5) Terminate the source system

Terminate the source database (PDMII E2 or RDB1 E2), and then terminate the source system.

The program termination procedure at the source system depends on the product specifications. For details, see the applicable manual.

(6) Terminate the target system

Terminate the target system.

To terminate a target system:

1. Terminate the data linkage product (Datareplicator) at the target system.
2. Terminate the target database (HiRDB).

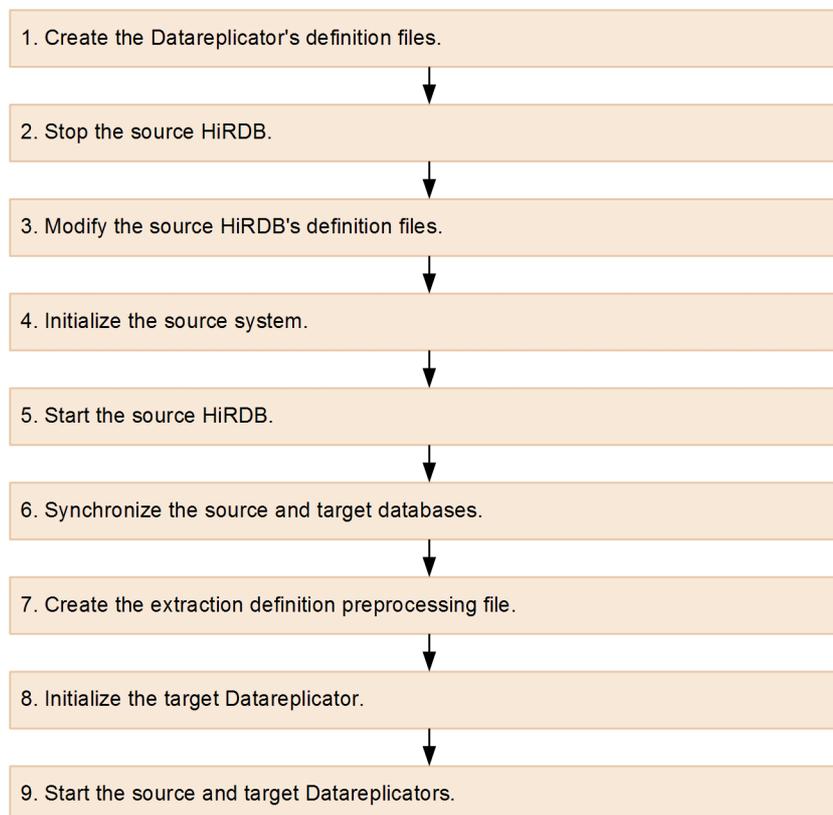
For details about target Datareplicator termination, see *6.6 Startup and termination of the target Datareplicator*. For details about HiRDB termination, see the *HiRDB Version 9 System Operation Guide*.

6.2 Initialization procedure at the environment configuration stage

You must initialize the Datareplicators when you start data linkage for the first time after you have set the Datareplicator configuration.

The following figure shows the general procedure.

Figure 6-3: Initialization procedure at the environment configuration stage



Note:

You must start the target database to perform the steps beginning with step 6 below.

The initialization procedure at the environment configuration stage shown in the figure is explained below. The numbers correspond to the numbers in the figure.

1. Create the following Datareplicator definition files:
 - Extraction system definition file
 - Transmission environment definition file
 - Extraction environment definition file
 - Extraction definition file
 - Duplexing definition file (for the source Datareplicator) (optional)
 - Import system definition file
 - Import environment definition file
 - Import definition file
 - Duplexing definition file (for the target Datareplicator) (optional)
2. Terminate the source HiRDB.

No.	Task	Execution command	Check item
1	Terminate the source HiRDB normally.	pdstop	Check that the KFPS01850-I message has been output to the syslog file (to the event log in Windows).

Note:

If the HiRDB is not terminated normally, changes to HiRDB definition files will not take effect.

3. Modify the source HiRDB's definition files listed below.

For details, see *5.6 Source HiRDB definition*.

- pd_rpl_init_start in the system common definition
(We recommend that you specify Y in the pd_rpl_init_start operand.)
- pd_log_rpl_no_standby_file_opr in the system common definition
- pd_rpl_hdepath in the unit control information definition

4. Initialize the source system.

No.	Task	Execution command	Check item
1	Initialize the source Datareplicator.	<code>hdestart -i</code> (Enter Y in the response message)	Check that the <code>KFRB00504-I</code> message has been output to <code>msterrfile1</code> or <code>msterrfile2</code> . If the <code>KFRB00504-I</code> message has not been output or <code>msterrfile1</code> and <code>msterrfile2</code> have not been created, a definition analysis error might have occurred or file creation might have failed. Determine the nature of the error by referencing the syslog file (event log in Windows), take appropriate action, and then re-initialize the source Datareplicator.

5. Start the source HiRDB.

Once you have started the source HiRDB, do not perform any updating application on the source database until you have completed step 8.

No.	Task	Execution command	Check item
1	Start the source HiRDB.	<code>pdstart</code>	Check that the <code>KFPS05210-I</code> message has been output to the syslog file (event log in Windows).
2	Start HiRDB Datareplicator linkage. Perform this task if nothing or <code>N</code> is specified in the <code>pd_rpl_init_start</code> operand in HiRDB's system common definition.	<code>pdrplstart</code>	Check that the <code>KFPS05140-I</code> message has been output to the standard output.
3	Verify that HiRDB is ready for data linkage with Datareplicator.	<code>pdls -d rpl -j</code>	Check the displayed results. [#]

#

As shown in the example below, verify that (1) and (2) are both Y in the displayed results. Also verify that Gen No and Block No indicated as (3) and (4), respectively, are not 0.

6. Operation

```

SYSTEMID      : HRD1(183346)
Data replication : Y ... (1)
UNITID       : unt1(183346)
Data replication : Y ... (2)
SERVER NAME  : sds01
Extract Database : Y
Extract Status : C
System Log Extract Point :
Run ID   Group   Gen No.  BLock No.
4740e7a9 log24   18       2dd8     ... (3)
System Log Sync Info :
Run ID   Group   Gen No.  BLock No.
4740e7a9 log24   18       2dba     ... (4)

```

6. Synchronize data in the source and target databases.

You synchronize the data in the source and target databases by using a program such as HiRDB Datareplicator to copy data from a source table to a target table in the batch mode (initial data creation).

7. Create the extraction definition preprocessing file.

No.	Task	Execution command	Check item
1	Create the extraction definition preprocessing file.	<code>hdeprep -f extraction-definition-file</code>	Check that the KFRB04500-I message has been output to the standard output.

Note:

Specify options in the hdeprep command as necessary.

8. Initialize the target Datareplicator.

No.	Task	Execution command	Check item
1	Initialize the target Datareplicator.	<code>hdsstart -i -f -q</code>	Check that the KFRB04216-I message has been output to the standard output (Event Viewer in Windows). If the KFRB04216-I message has not been output, a definition analysis error might have occurred or file creation might have failed. Determine the nature of the error by referencing the standard output (Event Viewer in Windows), take appropriate action, and then re-initialize the target Datareplicator.

Note:

When you initialize the target Datareplicator for the first time, we recommend that you execute the `hdsstart` command with the `-f` option specified. You can also specify other options as necessary.

9. Start the source and target Datareplicators.

No.	Task	Execution command	Check item
1	Start the target Datareplicator.	<code>hdsstart</code>	Check that the <code>KFRB00100-I</code> message has been output to <code>errfile1</code> or <code>errfile2</code> .
2	Start the source Datareplicator.	<code>hdestart</code> [#]	Check that the <code>KFRB00502-I</code> message has been output to <code>msterrfile1</code> or <code>msterrfile2</code> .

#

Specify options in the `hdestart` command as necessary.

When using the import transaction synchronization facility

If you will be using the import transaction synchronization facility, after you have finished steps 1 through 9, execute a synchronous event to activate all processes in the target Datareplicator's synchronous import group.

No.	Task	Execution command	Check item
1	Execute a synchronous event.	<code>hdeevent</code> [#] <code>-n synchronous-event-code</code>	<ul style="list-style-type: none"> • Check that no messages have been output to the standard error output. • Check that the <code>KFRB03009-I</code> message has been output to the target system's error information file and that <code>reason</code> is <code>SYNC EVENT</code>.

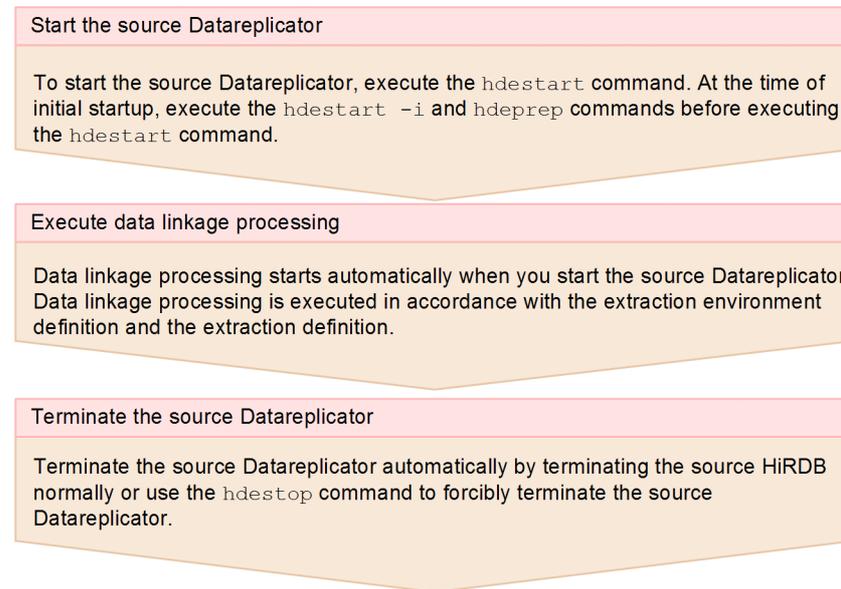
#

Specify options in the `hdeevent` command as necessary.

6.3 Startup and termination of the source Datareplicator

The following figure shows the procedures for starting and terminating the source Datareplicator.

Figure 6-4: Procedures for starting and terminating the source Datareplicator



6.3.1 Starting the source Datareplicator

This subsection explains the source Datareplicator's start method and start modes.

For details about the extraction processing start method, see *4.6.5 Designing the extraction processing start method*.

(1) How to start the source Datareplicator

When you execute the `hdestart` command at the source system, the source Datareplicator starts in accordance with the extraction system definition, extraction environment definition, and transmission environment definition. In the Windows edition of Datareplicator, execute the command at the command prompt.

To initialize the source Datareplicator's environment, execute the `hdestart -i` command. To initialize the transmission environment only for a specific target, use the `hdestart -i -S` command to initialize the environment only for the corresponding target.

If you use HiRDB's standby-less system switchover (effects distributed) facility, you

can also use the `hdestart_n` command to execute partial start (to start extraction master processes and extraction node master processes separately).

For details, see the `hdestart` command in Chapter 7. *Command Syntax*.

In the Windows edition of Datareplicator, you can also use either of the following methods to start the source Datareplicator:

- Start the extraction service manually.
- Set the extraction service to start automatically when Windows starts.

To set the extraction service to start automatically when Windows starts:

1. In **Control Panel**, double-click the **Services** icon.
2. Verify that the node master process startup service is started.
3. Under **Service**, choose **HiRDB Datareplicator (Source Site)**.
4. In the **Startup Parameters** field, specify the startup options (options for the `hdestart` command).
5. Click the **Start** button.

Notes on initial startup by specifying the `-i` option

- When you execute initial start from the service, Datareplicator does not display a message asking whether files are to be initialized. Therefore, before you execute an initial start, check that HiRDB Datareplicator linkage is not underway at the source HiRDB.
- Because the source Datareplicator terminates once file initialization is completed, an error message might appear in the Service dialog box. In such a case, check the event viewer or error log file for the initialization completion message. If an initialization completion message has been issued, initial start executed normally.

Setting the extraction service to start automatically when Windows starts

The following explains the procedure for setting the extraction service to start automatically when Windows starts. In this case, you cannot specify any start options (options in the `hdestart` command).

To set the extraction service to start automatically when Windows starts:

1. In **Control Panel**, double-click the **Services** icon.
2. Verify that the node master process startup service is set to start automatically.
3. Under **Service**, choose **HiRDB Datareplicator (Source Site)**.
4. Click the **Startup** button.
5. In the Service dialog box, under **Startup Type**, choose **Automatic**.

6. Click the **OK** button.
7. Click the **Close** button.

The next time you start Windows, the extraction service will start automatically.

(2) Start modes for the source Datareplicator

There are four start modes for the source Datareplicator, as described below.

For details about the start modes, see the *hdestart* command in Chapter 7. *Command Syntax*.

Initial start

The initial start mode only initializes the following files; it does not actually start the source Datareplicator:

- Extraction information queue files
- Extraction master status file
- Extraction server status file
- Extraction master error information files
- Extraction node master error information files
- Extraction master trace files
- Extraction node master trace files
- Data linkage file

Partial initial start

The partial initial start mode initializes the transmission environments for specified destinations only, in accordance with the extraction system definition or the transmission environment definitions for the specified destinations (partial initialization). It does not change the transmission environments for any other destinations. This mode does not actually start the source Datareplicator.

Normal start

The normal start mode starts the source Datareplicator in accordance with the extraction system definition, extraction environment definition, and transmission environment definitions without inheriting the settings of the previous session.

Restart

When extraction or transmission processing in the previous session terminated with an error, the restart start mode starts the source Datareplicator with the same settings that were in effect for the previous session in order to guarantee conformity between source and target databases.

6.3.2 Terminate the source Datareplicator

This subsection explains the source Datareplicator's termination method and termination modes.

For details about the extraction processing stop method, see *4.6.6 Designing the extraction processing stop method*.

(1) How to terminate the source Datareplicator

When the source HiRDB terminates normally and `true` was specified in the `syncterm` operand in the extraction system definition, the source Datareplicator terminates automatically (normal termination).

If you execute the `hdestop` command at the source system, the source Datareplicator terminates (forced termination). In the Windows edition of Datareplicator, execute the command at the command prompt.

If you use HiRDB's standby-less system switchover (effects distributed) facility, you can also use the `hdestop_n` command to execute partial termination (to terminate extraction master processes and extraction node master processes separately).

To terminate the source Datareplicator, make sure that you use the `hdestop` command. If the source Datareplicator is terminated by using an OS command to terminate processes, the results might not be as expected.

In the Windows edition of Datareplicator, you can also use the following method to terminate the source Datareplicator (forced termination).

Terminate the extraction service manually:

1. In **Control Panel**, double-click the **Services** icon.
2. Under **Service**, choose **HiRDB Datareplicator (Source Site)**.
3. Click the **Stop** button.

If you terminate Windows while the source Datareplicator is running, the source Datareplicator is terminated forcibly.

(2) Termination modes for the source Datareplicator

There are two termination modes for the source Datareplicator.

For details about the termination modes, see the `hdestop` command in Chapter 7. *Command Syntax*.

Normal termination

When the source HiRDB terminates normally, the source Datareplicator terminates automatically. To terminate the source Datareplicator, specify `true` in the `syncterm` operand in the extraction system definition.

Forced termination

The source Datareplicator terminates upon completion of all extraction and transmission processing that was underway when the `hdestop` command was executed. When you terminate the source Datareplicator forcibly, conformity might be lost between the source and target databases. To guarantee database conformity in such a case, you must use the restart mode the next time you start the source Datareplicator.

6.4 Operation of the source Datareplicator

This section explains the source Datareplicator operating procedures.

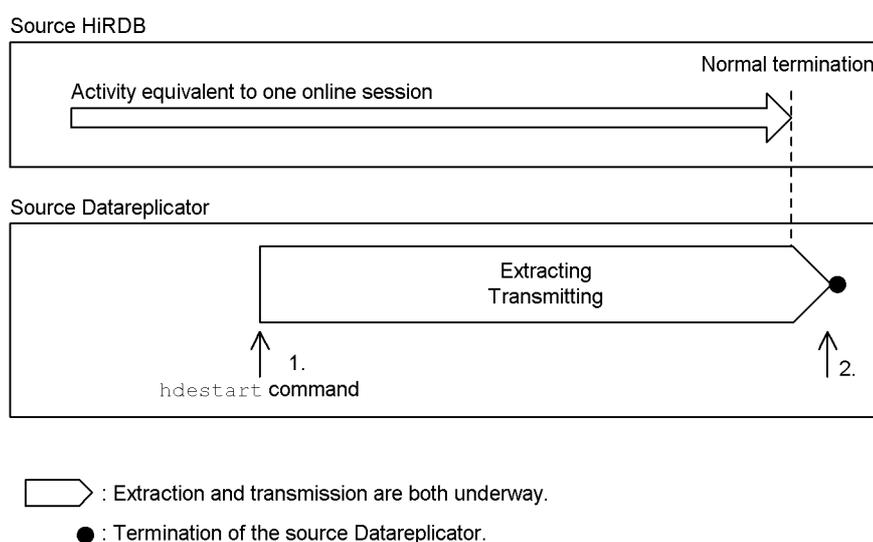
6.4.1 Handling of extraction processing

This section explains the extraction processing handling procedures.

(1) Starting the source Datareplicator by using the simultaneous start method

The simultaneous start method starts both extraction and transmission simultaneously when the source Datareplicator starts. Figure 6-5 shows an example of source Datareplicator operation using the simultaneous start method, and Table 6-1 explains the handling procedure and source Datareplicator processing when the simultaneous start method is used.

Figure 6-5: Example of source Datareplicator operation using the simultaneous start method



Note: The numbers in the figure (1 and 2) correspond to the numbers in the following table.

Table 6-1: Handling procedure and source Datareplicator processing (simultaneous start)

No.	Handling procedure	Source Datareplicator processing
1	Execute the <code>hdestart</code> command (for an initial startup, execute the <code>hdestart -i</code> and <code>hdeprep</code> commands before executing the <code>hdestart</code> command).	The source Datareplicator starts and begins extracting update information from the source HiRDB's system log file and sending it to the target system.
2	--	<p>When <code>true</code> is specified in the <code>syncterm</code> operand: The source Datareplicator terminates automatically when the source HiRDB terminates normally.</p> <p>When <code>false</code> is specified in the <code>syncterm</code> operand: The source Datareplicator does not terminate normally when the source HiRDB terminates normally. If you want to terminate the source Datareplicator while <code>false</code> is specified in the <code>syncterm</code> operand, first check the status with the <code>pdlis</code> or <code>hdestate</code> command, and then execute the <code>hdestop</code> command.</p>

Legend:

--: No action is needed.

(2) Starting the source Datareplicator by using the transmission delay start method

The transmission delay start method starts only extraction processing when the source Datareplicator starts and starts transmission processing when start of transmission is requested by the `hdestart -s` command.

To start the source Datareplicator by using the transmission delay start method:

1. At the source system, execute the `hdestart -e` command to start the source Datareplicator.

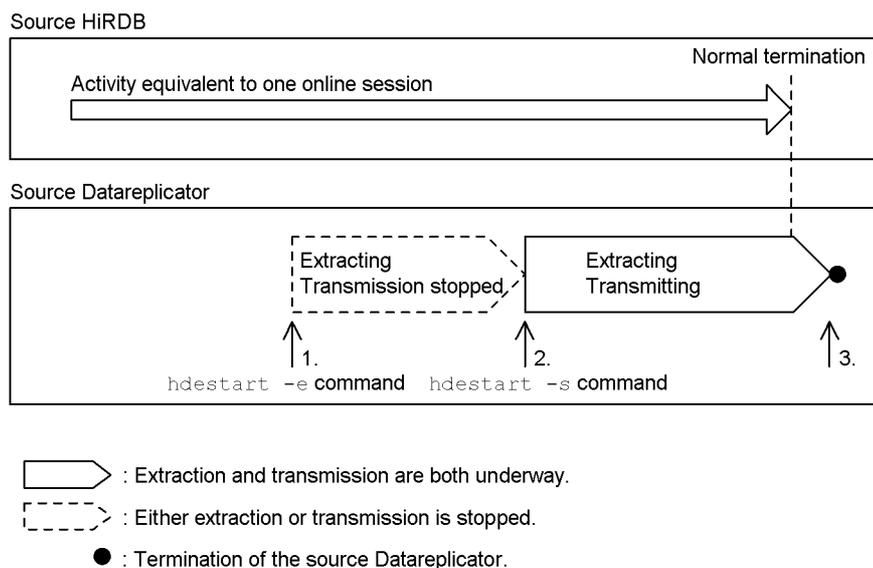
Only extraction processing starts.

2. At the source system, execute the `hdestart -s` command.

Transmission processing starts. You can also start transmission processing for a specific destination only by specifying the target identifier in the `hdestart -s` command.

Figure 6-6 shows an example of source Datareplicator operation using the transmission delay start method, and Table 6-2 explains the handling procedure and source Datareplicator processing when the transmission delay start method is used.

Figure 6-6: Example of source Datareplicator operation using the transmission delay start method



Note: The numbers in the figure (1 through 3) correspond to the numbers in the following table.

Table 6-2: Handling procedure and source Datareplicator processing (transmission delay start)

No.	Handling procedure	Source Datareplicator processing
1	Execute the <code>hdestart -e</code> command.	The source Datareplicator starts and begins extracting update information from the source HiRDB's system log file.
2	Execute the <code>hdestart -s</code> command before the extraction process detects the source HiRDB's normal termination. If the transmission process is not started before the extraction process detects the source HiRDB's normal termination, update information in the extraction information queue file will not be sent to the target system.	The source Datareplicator starts sending update information to the target system.

No.	Handling procedure	Source Datareplicator processing
3	--	<p>When <code>true</code> is specified in the <code>syncterm</code> operand: The source Datareplicator terminates automatically when the source HiRDB terminates normally.</p> <p>When <code>false</code> is specified in the <code>syncterm</code> operand: The source Datareplicator does not terminate normally when the source HiRDB terminates normally. If you want to terminate the source Datareplicator while <code>false</code> is specified in the <code>syncterm</code> operand, first check the status with the <code>pdls</code> or <code>hdestate</code> command, and then execute the <code>hdestop</code> command.</p>

Legend:

--: No action is needed.

(3) Starting the source Datareplicator by using the extraction delay start method

The extraction delay start method starts only transmission processing when the source Datareplicator starts and starts extraction processing when start of extraction is requested by the `hdestart -e` command.

To start the source Datareplicator by using the extraction delay start method:

1. At the source system, execute the `hdestart -s` command.

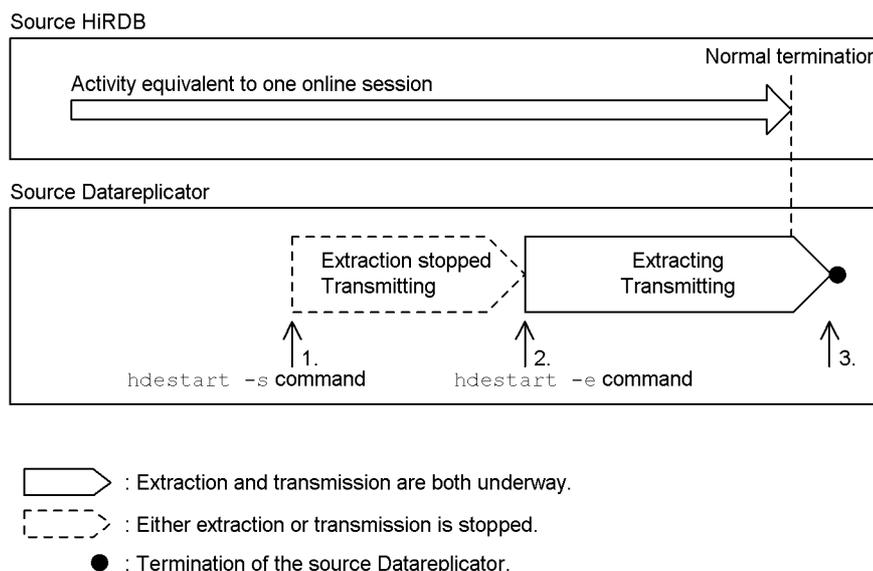
Only transmission processing starts. You can also start transmission processing for a specific destination only by specifying the target identifier in the `hdestart -s` command.

2. At the source system, execute the `hdestart -e` command.

Extraction processing starts.

Figure 6-7 shows an example of source Datareplicator operation using the extraction delay start method, and Table 6-3 explains the handling procedure and source Datareplicator processing when the extraction delay start method is used.

Figure 6-7: Example of source Datareplicator operation using the extraction delay start method



Note: The numbers in the figure (1 through 3) correspond to the numbers in the following table.

Table 6-3: Handling procedure and source Datareplicator processing (extraction delay start)

No.	Handling procedure	Source Datareplicator processing
1	Execute the <code>hdestart -s</code> command.	The source Datareplicator starts and begins transmitting update information to the target system.
2	Execute the <code>hdestart -e</code> command.	The source Datareplicator starts extracting update information from the source HiRDB's system log file.
3	--	When <code>true</code> is specified in the <code>syncterm</code> operand: The source Datareplicator terminates automatically when the source HiRDB terminates normally. When <code>false</code> is specified in the <code>syncterm</code> operand: The source Datareplicator does not terminate normally when the source HiRDB terminates normally. If you want to terminate the source Datareplicator while <code>false</code> is specified in the <code>syncterm</code> operand, first check the status with the <code>pdl</code> s or <code>hdestate</code> command, and then execute the <code>hdestop</code> command.

Legend:

--: No action is needed.

(4) Restarting only the part of the transmission processing that is in error status

To restart only the part of the transmission processing that is in error status:

1. At the source system, execute the `hdestate` command to check the transmission processing that is in error status.
2. At the source system, enter the `hdestate -s` command specifying the target identifier of the transmission processing in error status in order to determine the reason for the error.
3. Eliminate the cause of the error, and then execute the `hdestart -s` command specifying the target identifier of the transmission processing that is to be started.

The transmission processing with the specified target identifier starts, inheriting the previous mode.

(5) Adding a transmission target for the source Datareplicator

To add a transmission target, you must terminate data linkage, and then change the source or target Datareplicator definition. The transmission target whose definition has been changed must be initialized, but you can also execute partial initialization.

For details about how to add a transmission target, see *6.5.6 Source HiRDB handling procedure*.

6.4.2 Handling of the files used with the source Datareplicator

This section explains the procedures for handling the files used with the source Datareplicator.

For details about the contents of the files, see *3.2.2 Files and processes used during extraction processing*.

For details about the preparation of the files, see *4.6.2 Preparation of the files used with the source Datareplicator*.

Note: Do not replace files, regardless of whether Datareplicator is running or stopped. If any of these files is replaced, Datareplicator might malfunction.

(1) Extraction system definition file handling

For details about how to handle the extraction system definition file, see *5.2.2 Modifying defined information*.

(2) Extraction environment definition file handling

For details about how to handle the extraction environment definition file, see *5.3.2 Modifying defined information*.

(3) Transmission environment definition file handling

For details about how to handle the transmission environment definition file, see 5.4.2 *Modifying defined information*.

(4) Extraction definition file handling

For details about how to handle the extraction definition file, see 5.5.2 *Modifying defined information*.

(5) Extraction definition preprocessing file handling

The following explains how to handle the extraction definition preprocessing file.

Converting the extraction definition file again

If you have modified definitions for a table subject to extraction processing at the source HiRDB or you have modified the source Datareplicator's extraction definition, you must convert the extraction definition file to the internal format again.

To convert the extraction definition file to the internal format:

1. Terminate the source Datareplicator normally.
2. Modify the definitions for the table subject to extraction processing or the extraction definition.
3. At the source system, execute the `hdeprep` command.
4. Start the source Datareplicator normally.

(6) Extraction information queue file handling

The following explains how to handle the extraction information queue files. For details about the handling procedure when an extraction information queue file is full, see 9.1.2 *Error handling methods*.

(a) Modifying a filename, the file size, or the number of files

When using a UNIX regular file or Windows file:

1. Terminate the source system.
2. Terminate the source Datareplicator normally.
3. Use a text editor to modify the corresponding operands in the extraction environment definition.
4. Execute initial start on the source Datareplicator.
5. Start the source Datareplicator normally.
6. Start the source system.

When using a UNIX character special file:

1. Terminate the source system.
2. Terminate the source Datareplicator normally.
3. Use an OS command to re-create the extraction information queue file in character special file format.

Note:

Perform this step only when a status file size or the disk arrangement has changed for a reason such as the addition or removal of HiRDB servers.

4. Use a text editor to modify the corresponding operands in the extraction environment definition as appropriate for the created extraction information queue file.

Note:

Perform this step only when a status file size or the disk arrangement has changed for a reason such as the addition or removal of HiRDB servers.

5. Execute initial start on the source Datareplicator.
6. Start the source Datareplicator normally.
7. Start the source system.

(b) Command for changing the organization of extraction information queue files (hdemodq command)

The `hdemodq` command enables you to change the organization of the extraction information queue files without having to initialize the source Datareplicator. The following table describes the operations supported by the `hdemodq` command:

hdemodq command operation	Description
Displaying information about the extraction information queue files	You can display information about the extraction information queue files in offline mode.
Registering additional extraction information queue files	To avoid a file full status, you can use this command to add extraction information queue files.
Releasing registered extraction information queue files	You can delete any extraction information queue file that is no longer needed.

Prerequisites for executing the `hdemodq` command

You can execute the `hdemodq` command under the following conditions:

- The source Datareplicator is inactive.

- The `HDEPATH` environment variable has been set.
- The extraction information queue file whose organization is to be changed is located at the node where the command is executed.
- The user executing this command also executes the extraction node master process in a UNIX system (specified in `inetd.conf`) or has the Administrators privilege in a Windows system. General users can execute this command only when the `-l` option is specified in the `hdemodq` command (to display file information).

You can execute multiple `hdemodq` commands concurrently at the same source system only to display information. Such concurrent command execution is not supported for registration of additional files or release of file registration (if such an attempt is made, all but one of the commands issue the `KFRB09302-E` message and terminate in an error).

Displaying extraction information queue file information

This command enables you to display information about the extraction information queue files at the standard output. For details about the display format, see the `hdemodq` command in Chapter 7. *Command Syntax*.

Registering additional extraction information queue files

This command enables you to initialize as much space as needed for an extraction information queue file and register the file in Datareplicator. This additional registration is not applicable when a shortage of disk space (`DISK FULL`) is caused by write operations on an extraction information queue file. You can add a maximum of 16 extraction information queue files.

The types of extraction information queue files that can be added are UNIX regular files and character special files (and Windows files). Extraction information queue files cannot be added to the Datareplicator file system area.

An added extraction information queue file must be the same size as the existing extraction information queue files.

Releasing registered extraction information queue files

This command enables you to release extraction information queue files that have been registered in Datareplicator. This command function is applicable only to those extraction information queue files whose untransmitted extraction information status is `e`. The `hdemodq` command only releases a file's registration; it does not actually delete the extraction information queue file.

You can reduce the number of unreleased extraction information queue files to a minimum of 2. When there are only two extraction information queue files registered in Datareplicator, you can no longer perform extraction information queue file registration release.

The types of extraction information queue files whose registration can be released are UNIX regular files and character special files (and Windows files), as well as extraction information queue files in the Datareplicator file system area.

Note:

Note the following points about using the `hdemodq` command to change the organization of extraction information queue files:

- Backing up the extraction server status file

The `hdemodq` command updates the extraction server status file because it changes the status of extraction information queue files. To be prepared for the possibility of command execution errors, always back up the extraction server status file before executing the command. If your extraction server status file is stored in the Datareplicator file system area, back up the entire Datareplicator file system area. For details about how to back up the extraction server status file, see *6.4.2(7)(b) Backing up the extraction server status file*.

- Initialization after an organization change

If you initialize the environment of the source Datareplicator by executing the `hdestart -i` command after changing the organization of the extraction information queue files with the `hdemodq` command, the environment is also re-constructed according to the extraction environment definition. Therefore, the system ignores the change made to the organization of the extraction information queue files by the `hdemodq` command.

(7) Extraction status file handling

The following explains how to handle the source Datareplicator's status files (extraction master status file and extraction server status file).

(a) Initializing an extraction status file

To initialize an extraction status file (extraction master status file or extraction server status file)

When using a UNIX regular file or Windows file:

1. Terminate the source system.
2. Terminate the source Datareplicator normally.
3. Execute initial start on the source Datareplicator.
4. Start the source Datareplicator normally.

Start the source system.

When using a UNIX character special file:

1. Terminate the source system.

2. Terminate the source Datareplicator normally.
3. Use an OS command to delete the previous status file.
Perform this step only when a status file size or the disk arrangement has changed for a reason such as the addition or removal of HiRDB servers.
4. Use an OS command to re-create the status file in character special file format.
Perform this step only when a status file size or the disk arrangement has changed for a reason such as the addition or removal of HiRDB servers.
5. Execute initial start on the source Datareplicator.
6. Start the source Datareplicator normally.
7. Start the source system.

(b) Backing up the extraction server status file

Back up the extraction server status file in the following cases:

- You will be using the `hdemodq` command to change the organization of extraction information queue files.
- The extraction information queue file has become full.
- You will be using the facility for recovering the extraction information queue file.

The following table shows the commands used to back up the extraction server status file.

Table 6-4: Commands used to back up the extraction server status file

Type of extraction server status file	Command to be used	Example of command execution
UNIX regular file	<code>cp</code>	<code>cp \$HDEPATH/sts_sds01 sts_sds01_backup</code>
UNIX character special file	<code>dd</code>	<code>dd if="\$HDEPATH/sts_sds01" of=sts_sds01_backup bs=1024 count=number-of-kilobytes-in-sts_sds01</code>
Windows file	<code>copy</code>	<code>copy "%HDEPATH%\sts_sds01" sts_sds01_backup</code>

Note 1:

In this example, the name of the extraction server status file is `sts_sds01`. For details about each command, see the applicable OS documentation.

Note 2:

To recover the extraction server status file from its backup, execute the command with the extraction server status file name and the backup file name swapped.

(8) Extraction error information file handling

The following explains how to handle the extraction error information files (extraction master error information files and extraction node master error information files).

(a) Changing the maximum size

To change the maximum size of the extraction error information files:

1. Terminate the source system.
2. Terminate the source Datareplicator normally.
3. Use a text editor to modify the appropriate operand in the extraction system definition.
4. Start the source Datareplicator normally.
5. Start the source system.

(b) Saving an extraction error information file

When an error information file subject to data accumulation becomes full, file swapping occurs. Datareplicator re-creates the next error information file so that it can be swapped in.

To save the contents of an error information file before it is re-created:

1. Use an OS command to check the most recent update dates and times of the error information files.
2. Use an OS command to copy the error information file with the earlier update date and time into a new file under any name.

Datareplicator issues a message (KFRB00051-I or KFRB00052-I) whenever error information files are to be swapped or whenever an error information file is to be closed during Datareplicator operation; when such a message is issued, make a backup copy if necessary.

(c) Outputting error information to the standard output

When the error information file being used for data accumulation becomes full, file swapping occurs. Datareplicator re-creates the next error information file that is to be swapped in.

To output the contents of the error information file to the standard output before the file is re-created:

1. Use an OS command to check the most recent update dates and times of the error information files.
2. Use an OS command to output the extraction master error information file with the older update date and time to the standard output.

(d) Examples of error information file output

Figure 6-8 shows an example of the output from an extraction master error information file, and Figure 6-9 shows an example of the output from an extraction node master error information file.

Figure 6-8: Example of output from an extraction master error information file

```

*****
1. Fri Mar 29 21:15:39 2002 process: hdemaster(1324) function:hde_mst_main
2. errorcode: KFRB00701-E .. 00 Error occurred on node-master process,
   hostname = host01, server name = bes01, senderid = snd01, message number = 00703
3. information:
   *****
   Fri Mar 29 21:16:21 2002 process: hdemaster(1301) function:hde_com_main
   errorcode: KFRB00601-E .. 00 Create socket failed, error, errno = 12.
   information:

```

1. Fri...2002, process, function
 Fri...2002: Day of week, month, date, time (*hh:mm:ss*), and year when error occurred.
 process: Name and number of internal process resulting in the error.
 function: Name of internal function resulting in the error.
2. errorcode ... message number=00703
 This is the error message. For details about the message, see Chapter 10. *Messages*.
3. information
 This is detailed information about the error.

Figure 6-9: Example of output from an extraction node master error information file

```

*****
1. Fri Mar 29 21:15:39 2002 process: hdcapture(1324) function: hds_com_read
2. serverid: SERVER01 senderid:SENDER01
3. errorcode: EFRB05011-E aa 00 HiRDB overwrites unextracted system-log.
4. information:
*****
Fri Mar 29 21:16:21 2002 process: hdesender(1301) function: hds_com_main
serverid: SERVER01 senderid:SENDER02
errorcode: EFRB02013-E aa 00 Socket option set error,  errno = 12.
information:

```

1. `Fri...2002, process, function`
`Fri...2002:` Day of week, month, date, time (*hh:mm:ss*), and year when error occurred.
`process:` Name and number of internal process resulting in the error.
`function:` Name of internal function resulting in the error.
2. `serverid, senderid`
`serverid:` Name of server resulting in the error.
`senderid:` Transfer identifier resulting in the error.
3. `errorcode ... system-log`
This is the error message. For details about the message, see Chapter 10. *Messages*.
4. `information`
This is detailed information about the error.

(e) Output destinations other than files

Datareplicator outputs the contents of error information files to the syslog file so that the information can be used to prevent the types of errors reported in the error information files and achieve automatic operation.

To output the contents of error information files to the syslog file, you must set the `syslogout` operand to `true` in the extraction system definition.

The following table shows the output destination for the contents of an extraction master error information file.

Table 6-5: Output destinations for the contents of an extraction master error information file

Status of the extraction master error information file	Output destination	
	syslog file	Extraction master error information file
Normal (output enabled)	If true	Y

Status of the extraction master error information file	Output destination	
	syslog file	Extraction master error information file
Error (output disabled)	If true	--

Y: Output.

If true: Output if `true` is specified in the `syslogout` operand in the extraction system definition.

--: Not output.

(9) Activity trace file handling

You can use the `hdstrcredit` command to view and edit the activity trace files (extraction master trace files and extraction node master trace files). For details about how to use the `hdstrcredit` command, see the *hdstrcredit* command in Chapter 7. *Command Syntax*.

(10) Data linkage file handling

The following explains how to handle the data linkage file.

Initializing the data linkage file

When using a UNIX regular file or Windows file:

1. Terminate the source system.
2. Terminate the source Datareplicator normally.
3. Execute initial start on the source Datareplicator.
4. Start the source Datareplicator normally.
5. Start the source system.

When using a UNIX character special file:

1. Terminate the source system.
2. Terminate the source Datareplicator normally.
3. Use an OS command to delete the previous data linkage file.
4. Use an OS command to re-create the data linkage file in character special file format.
5. Execute initial start on the source Datareplicator.
6. Start the source Datareplicator normally.
7. Start the source system.

(11) Command log file handling

The following explains how to handle a command log file.

The command log files contain a record of when Datareplicator commands were executed. The command log files are created automatically when Datareplicator starts. You can view a command log file at any time to check the execution history of commands. For details about the information that is output to the command log file, see *Overview of commands* in 7. *Command Syntax*.

The following figure shows an example of the contents of a command log file.

Figure 6-10: Example of command log file contents

```
Fri Aug 18 09:36:52 2006 pid=596 SYSTEM hdestart : END : status=0, arg= -i.
Fri Aug 18 09:37:15 2006 pid=912 administrator hdeprep : END : status=0, arg= -f extfile.
Fri Aug 18 09:40:18 2006 pid=952 SYSTEM hdestart : END : status=0, arg=.
```

6.4.3 Notes on handling the source Datareplicator

This section provides important information about handling the source Datareplicator.

(1) Notes on update information

- If a table at a target system is updated from multiple source systems, the order of the update processing is unpredictable. If you need to control the order of update processing, you must do so by handling the processing appropriately. Suppose that data linkage is established from two source systems, A and B, to table T1 at a target system, and that a table row corresponding to source system A's mapping key value 100 is updated. Then a table row corresponding to source system B's mapping key value 100 is updated. In this case, the contents of the table row corresponding to key value 100 at the target system might not be the same as that of the corresponding row at source system B.
- Before you modify the extraction definition at the source system, you must ensure that all update information has been sent from the source system to the target Datareplicators. If you modify the extraction definition while some update information still remains untransmitted at the source Datareplicator, import processing might result in an error. In such a case, a target Datareplicator might not be able to recover the error, necessitating use of the HiRDB Dataextractor to re-create the target database.
- The source Datareplicator sends the update information for transactions that have been completed within the transmission interval to the target system, where the update information is imported in the order the transactions were committed. Therefore, a transaction that is not completed within the transmission interval might be sent to the target system after a transaction that occurs later. In such a case, the order of update processing might not match between the source and

target databases.

- If the source HiRDB is a parallel server, the source Datareplicator extracts and transmits update information for all back-end servers in parallel. Therefore, the order of update processing among the back-end servers is not predictable. Conformity between the source and target databases is guaranteed when all update information has been extracted from the system log files at all back-end servers, sent, and imported into the target database.

(2) Notes on initial start and partial initial start

If you start the source system in the initialized status, start the target Datareplicator with the `hdsstart -i` or `hdsstart -i -D` command. If you start the source Datareplicator using the partial initial start mode, use the `hdsstart -i` or `hdsstart -i -D` command to start the target Datareplicator corresponding to the specified destination. If you start only the source system in the initialized status, the KFRB02003-E error results (detail code 5).

If you specified `nocheck` in the `extract_init` operand in the import environment definition, there is no need to start the target Datareplicator in the initial start or partial initial start mode.

(3) Notes on event codes

Even if an event code sent from the source system is undefined in the import environment definition, there is no effect on the import processing; the target Datareplicator still recognizes it as an event.

(4) Notes about the HDE_BIN_COL_MAXLEN environment variable

If you specify the `HDE_BIN_COL_MAXLEN` environment variable (in kilobytes) in the source Datareplicator, you can perform data linkage on BLOB-type columns that have a definition length of 2 GB or greater but whose actual data is small without having to redefine the table. If the source Datareplicator detects a BLOB-type column that is longer than the definition length specified in the `HDE_BIN_COL_MAXLEN` environment variable, the source Datareplicator performs the following processing:

- Replaces the data in the corresponding BLOB-type column with the null value, and then imports it to the target Datareplicator.
- Outputs information about mapping keys from the update information resulting from the null value replacement to a file.

The following explains the file that is output.

File name and output destination:

The file is created under the name `warn_keyinfo.BES-name.target-name` in the source Datareplicator directory (directory specified in the `HDEPATH` environment variable). If the source database is in a HiRDB/Parallel Server, the file is created in the source Datareplicator directory on the machine that contains

the back-end server in which is located the BLOB-type column that exceeds the definition length specified in the HDE_BIN_COL_MAXLEN environment variable.

Contents of the file

The following shows an example of the contents that are output to the file:

```

Skip info [SQL : insert]
Table name : USR1.TBL1
Column name : C1
  [DATA : length=4]
    0001                                *....*
Column name : C2
  [DATA : length=24]
    30303030303030303030303030303030 *0000000000000000*
    3030303030303031                    *00000001*

```

Note:

- Delete this file manually before the source Datareplicator starts. If the file then exists when the source Datareplicator operation has terminated, it means that a BLOB-type column that is longer than the definition length specified in the HDE_BIN_COL_MAXLEN environment variable was detected.
- Information about the mapping key columns is output during transmission processing. If an error occurs during transmission processing, information about the mapping key columns that has been output once might be output again when transmission processing is restarted.

6.5 Handling of the source HiRDB

This section explains the handling of the source HiRDB when data is extracted from a HiRDB database.

6.5.1 Starting, stopping, and cancelling HiRDB Datareplicator linkage

(1) *Starting and terminating HiRDB Datareplicator linkage*

The timing with which HiRDB Datareplicator linkage starts is explained below; it terminates when the source HiRDB terminates:

- Specification of `pd_rpl_init_start=Y` in the source HiRDB's system common definition

To have HiRDB Datareplicator linkage resume following a normal startup, specify `Y` in the `pd_rpl_init_start` operand in HiRDB's system common definition.

When `pd_rpl_init_start=Y` is specified, HiRDB Datareplicator linkage inherits the output status of the data linkage information in the system log from the previous source HiRDB and the system log extraction status at the source Datareplicator, regardless of whether the source HiRDB is being started or restarted.

In the following cases, the source HiRDB initializes the extraction status and starts HiRDB Datareplicator linkage without inheriting the previous status, so in these cases you must re-create the target database before restarting HiRDB Datareplicator linkage because conformity between the source and target databases has been lost:

- Forced normal start (`pdstart dbdestroy`)
- Database initialization start (`pdstart -i`)
- Execution of the `pdrplstart` command

If HiRDB Datareplicator linkage is started by the `pdrplstart` command, the previous status is inherited during a restart, but not during a normal start. You need to re-execute the `pdrplstart` command after starting HiRDB normally. Therefore, we recommend that you specify `Y` in the `pd_rpl_init_start` operand in the source HiRDB's system common definition.

(2) *Cancelling HiRDB Datareplicator linkage*

HiRDB Datareplicator linkage is cancelled by the following methods:

- Execution of the `pdrplstop` command

- Modification of the `pd_rpl_init_start` operand in the source HiRDB's system common definition (changing the specification from `Y` to `N` or omitting the `pd_rpl_init_start` operand)

When HiRDB Datareplicator linkage is cancelled, conformity between the source and target databases is lost, in which case you must re-create the target database before restarting HiRDB Datareplicator linkage. For details about how to execute the `pdrplstop` command, see *6.5.6 Source HiRDB handling procedure*.

The HiRDB Datareplicator linkage terminated by the `pdrplstop` command is inherited during a restart, but not during a normal start.

If you cancel data linkage by changing the `pd_rpl_init_start` operand, make sure that the value of the `pd_rpl_hdepath` operand is not changed.

6.5.2 Data linkage file

The source HiRDB and the target Datareplicator use a *data linkage file* to report the extraction status. If an error occurs in the data linkage file or the data linkage file is initialized, conformity between the source and target databases might be compromised, because the data linkage preexisting at the source Datareplicator is lost. In such a case, you must re-create the target database. For details about how to re-create a target database, see *6.5.6 Source HiRDB handling procedure*.

HiRDB checks the data linkage file when a synchronization point dump is collected. If the data linkage file has been initialized, the source HiRDB cancels HiRDB Datareplicator linkage and resumes its own processing at the source HiRDB.

6.5.3 Handling of the system log file

When the source HiRDB starts HiRDB Datareplicator linkage, it manages the data linkage status by adding to the system log file standby status the status indicating whether extraction of system log file information has been completed at the source Datareplicator. This status is called the *extraction status*. This section explains the following topics:

- Status of the system log file
- Extraction status
- Specifying the extraction status
- Releasing the extraction status
- Specification when the system log file cannot be swapped in because of its extracting status
- Action to be taken when the source HiRDB unit is terminated forcibly due to the extracting status
- Manipulating the system log file

(1) Status of the system log file

The table below shows the system log file's standby status when the HiRDB Datareplicator linkage facility is used. You can use HiRDB's `pdlogls` and `pdl` commands to check the status of the system log file.

Table 6-6: System log file standby status when the HiRDB Datareplicator linkage facility is used

Status	Remarks			
Standby	Swappable target	Overwrite enabled	System log file can be swapped in if all three statuses are true.	
		Unload completed		
		Extracting status [#]		
	Unswappable target	Overwrite disabled		System log file cannot be swapped in if any of the three statuses is true.
		Unload wait		
		Extracting status [#]		

#

This status is added only while HiRDB Datareplicator linkage is executing.

Note:

If the source Datareplicator's extraction processing is stopped for an extended period of time after HiRDB Datareplicator linkage started, HiRDB's system log file might have become full and HiRDB might be shut down. We recommend that you do not stop the source Datareplicator's extraction processing while HiRDB is running.

(2) Extraction status

There are two extraction statuses for the system log files:

- Extraction completed status

In this status, extraction processing has been completed at the source Datareplicator. When the source Datareplicator finishes reading all system log information from a system log file, that system log file is placed in extraction completed status and becomes eligible to be swapped in if it is also in the overwrite enabled and unload completed statuses.

- Extracting status

This is the status in which extraction processing has not yet been completed at the source Datareplicator. A system log file is placed in extracting status when the

system log information it contains has not been extracted or is being extracted by the source Datareplicator. While it is in extracting status, the system log file cannot be swapped in even if it is also in overwrite enabled and unload completed status. If a system log file in extracting status is initialized or placed forcibly in swappable status, conformity between the source and target databases is lost, because some of the update information for the source database cannot be imported into the target database. If this happens, you must re-create the target database.

The `pdlogls` command might output the current file group as being in the unextracting status for HiRDB internal control purposes even when there is no unextracted update information in the system log. To determine whether the system log actually contains unextracted update information, execute `pdls -d rpl -j`.

All update information in the system log has been extracted when the values of Run ID, Gen No, and Block No displayed as System Log Extract Point are all greater than the values of the same items that are displayed as System Log Sync info.

(3) Specifying the extraction status

When HiRDB Datareplicator linkage starts, the extraction status for a system log file is specified by the following methods:

- `pd_rpl_init_start=Y` in the source HiRDB's system common definition
- `pdrplstart` command

The source HiRDB checks the extraction status of each system log file when a synchronization point dump is collected. When the source Datareplicator completes extraction from a system log file, the file's status changes from extracting to extraction completed at the time of the next synchronization point dump. This means that a system log file is not placed in extraction completed status until a synchronization point dump occurs.

(4) Releasing the extraction status

A system log file's extraction status is released when it becomes eligible to be swapped in. Until then, the system log file is displayed as being in the extraction status when the `pdlogls` command executed. A system log file in this status is still subject to extraction processing by the source Datareplicator.

The extraction status is released forcibly in the following cases:

- Execution of the `pdrplstop` command.
- `pd_log_rpl_no_standby_file_opr=continue` is specified in the system common definition, but the system log file cannot be swapped in because it is not in extraction completed status.
- The `pd_rpl_init_start` operand is modified in the source HiRDB's system common definition (changing from Y to N or omitting the `pd_rpl_init_start`

operand).

- Execution of the `hdestart` command with the `-i` option specified.
- System log file in extracting status is initialized or its extraction status is changed forcibly.
- Data linkage file is initialized.

When the extraction status is released, conformity between the source and target databases might be compromised, because the HiRDB Datareplicator linkage preexisting at the source Datareplicator cannot be restored the next time HiRDB Datareplicator linkage is started. In such a case, you must re-create the target database.

(5) Specification when the system log file cannot be swapped in because of its extracting status

If none of the system log files can be swapped in, you can specify an action to be taken when any of the system log files is in extracting status. This specification is made in the source HiRDB's system common definition, as shown below:

```
pd_log_rpl_no_standby_file_opr=stop|continue
```

For details about the specification of `pd_log_rpl_no_standby_file_opr` in the system common definition, see *5.6 Source HiRDB definition*. For details about the handling procedure when a system log file cannot be swapped in, see *6.5.6 Source HiRDB handling procedure*.

Terminating the corresponding source HiRDB unit forcibly in order to continue data linkage (when stop is specified or `pd_log_rpl_no_standby_file_opr` is omitted)

To continue data linkage, specify forced termination of the HiRDB unit in the system common definition. The corresponding source HiRDB unit will be terminated forcibly when there is no file that can be swapped in due to the extracting status.

To restart a forcibly terminated source HiRDB unit, check that a log file that is being extracted has been transformed to another file by the source Datareplicator (that is, check that there is at least one file in extraction completed status), and then restart the source HiRDB unit. You can determine the extraction status of a system log file by first executing the `pdlogls` command to obtain the status of each system log file, and then executing the `pdls -d rpl -j` command to obtain the progress of extraction processing.

For details about how to cancel data linkage and continue processing only at the source HiRDB when the source HiRDB unit is terminated forcibly, see (6) below.

Cancelling data linkage and continuing processing only at the source HiRDB (when continue is specified)

To cancel data linkage and continue processing only at the source HiRDB, specify

cancellation of HiRDB Datareplicator linkage in the system common definition. The source HiRDB cancels HiRDB Datareplicator linkage at all back-end servers when there are no more files that can be swapped in due to the extracting status, and then continues its processing. The source HiRDB releases the extraction status of all system log files when there are no more files that can be swapped in. As a result, the source HiRDB can resume processing because a system log file previously in extracting status can now be swapped in.

When the extraction status is released, you must re-create the target database before restarting HiRDB Datareplicator linkage because conformity between the source and target databases is lost.

(6) Action to be taken when the source HiRDB unit is terminated forcibly due to the extracting status

If the source HiRDB unit is terminated forcibly in accordance with a specification in the system common definition because the system log file is in extracting status, you must take one of the following actions to continue processing at the source HiRDB:

- To continue data linkage

Start the source Datareplicator if it has terminated. If both the extraction and transmission processes are active, check the target Datareplicator's status at the destination and start it if it has terminated.

- To cancel data linkage and continue processing only at the source HiRDB

If the system log file cannot be placed in extraction completed status due to an error in the source or target Datareplicator, cancel HiRDB Datareplicator linkage and restart the source HiRDB in order to continue processing. In this case, you must take one of the following actions before restarting the source HiRDB:

- Change to `continue` the specification of the `pd_log_rpl_no_standby_file_opr` operand in the system common definition.
- Use the `pdlogchg` command to forcibly place all system log files in extraction completed status.

When you cancel HiRDB Datareplicator linkage, conformity between the source and target databases is lost, in which case you must re-create the target database before restarting HiRDB Datareplicator linkage. For details about how to cancel data linkage when the source HiRDB unit is terminated forcibly, see *6.5.6 Source HiRDB handling procedure*.

(7) Manipulating a system log file

The following topics concerning manipulation of system log files are discussed here, because they are especially important with respect to data linkage:

- Checking a system log file's extraction status (`pdlogls` command)

- Changing a system log file's extraction status forcibly (`pdlogchg` command)
- Deleting a system log file in extracting status (`pdlogrm` command)

For details about other manipulations, see the *HiRDB Version 9 System Operation Guide*.

Checking a system log file's extraction status (`pdlogls` command)

You can use the `pdlogls` HiRDB command to check a system log file's extraction status. If you are using the HiRDB Datareplicator linkage facility, the following statuses are added to the unload status of file groups, element files, and physical files:

a: Unload wait and extracting status

A system log file in unload wait status contains log information that needs to be unloaded. The system log file is in extracting status when extraction of the data linkage information by the HiRDB Datareplicator has not been completed.

u: Unload wait and extraction completed status

e: Unload completed and extracting status

-: Unload completed and extraction completed status

A system log file's extraction status does not change when the corresponding unit becomes inactive at the source HiRDB. Even if extraction has been completed by the source Datareplicator, the `pdlogls` command displays the corresponding system log file as being in extracting status. Therefore, to check a system log file's extraction status while the corresponding unit is inactive, you must use the `pdls` HiRDB command to check the extraction status of the source Datareplicator.

Changing a system log file's extraction status forcibly (`pdlogchg` command)

If a system log file cannot be placed in extraction completed status due to an error at the source or target Datareplicator, you can use the `pdlogchg` HiRDB command to change the system log file's extraction status forcibly in order to continue processing at the source HiRDB. If you specify the `-R` option in the `pdlogchg` command, the system log file changes from extracting status to extraction completed status.

When you change the system log file's extraction status forcibly, conformity between the source and target databases is lost. In such a case, cancel HiRDB Datareplicator linkage, re-create the target database, and then restart HiRDB Datareplicator linkage. For details about the handling procedure during execution of the `pdlogchg` command, see *6.5.6 Source HiRDB handling procedure*. For details about how to specify the `-R` option in the `pdlogchg` command, see *7. Command Syntax*.

Deleting a system log file in extracting status (`pdlogrm` command)

To forcibly delete a system log file in extracting status, execute the `pdlogrm` HiRDB command with the `-u` option specified. You cannot delete a system log file that is in extracting status without specifying the `-u` option. When you delete a system log file

in extracting status, conformity between the source and target databases is lost, in which case you must re-create the target database before starting HiRDB Datareplicator linkage.

6.5.4 Processing at the source HiRDB and source Datareplicator depending on the specification of extsuppress in the extraction environment definition

Processing at the source HiRDB and source Datareplicator depends on the specification of `extsuppress` in the extraction environment definition. The following table shows the processing at the source HiRDB and source Datareplicator depending on the specification of `extsuppress` in the extraction environment definition.

Table 6-7: Processing at the source HiRDB and source Datareplicator depending on the specification of extsuppress in the extraction environment definition

Type of HiRDB server	Specification at the source Datareplicator		Processing at the source HiRDB	Processing after starting the source Datareplicator
	extsuppress in the extraction environment definition	Specification in the extraction definition		
MGR FES DS	N/A	N/A	N/A	N/A
BES SDS	true	Subject to extraction	Does not specify the extraction status.	Stops processing at this server after issuing a message indicating inconsistent definition. See (1) Handling procedure when true is specified, but the definition to extract the table is to be changed.
		Not subject to extraction	N/A	Stops processing at this server after issuing a message indicating that the table is not subject to extraction processing.
	false	Subject to extraction	Specifies the extraction status.	Starts data linkage processing.

Type of HiRDB server	Specification at the source Datareplicator		Processing at the source HiRDB	Processing after starting the source Datareplicator
	extsuppress in the extraction environment definition	Specification in the extraction definition		
		Not subject to extraction	Before starting the source Datareplicator: Specifies the extraction status. After starting the source Datareplicator: Releases the extraction status.	Stops processing at this server after issuing a message indicating that the table subject to extraction processing is not found. See (2) <i>Notes on a server that contains no table subject to extraction when false is specified.</i>

MGR: System manager

FES: Front-end server

DS: Dictionary server

BES: Back-end server

SDS: Single server

N/A: Not applicable

Subject to extraction: When the corresponding server contains the table specified as being subject to extraction in the extraction definition

Not subject to extraction: When the corresponding server does not contain the table specified as being subject to extraction in the extraction definition

(1) Handling procedure when true is specified, but the definition to extract the table is to be changed

When `true` is specified and the source Datareplicator processing has stopped due to inconsistent definition, and you want to change the definition to extract the table, modify the extraction environment definition, re-initialize the source Datareplicator (by executing the `hdestart -i` command), and then start the source Datareplicator.

If you start HiRDB Datareplicator linkage at the source HiRDB before re-initializing the source Datareplicator, conformity between the source and target databases is lost.

In such a case, cancel HiRDB Datareplicator linkage, re-create[#] the target database on the basis of the source database, and then re-initialize the source Datareplicator.

#

If the source HiRDB's table subject to data linkage has not been updated, there is no need to re-create the target database. Stop HiRDB Datareplicator linkage, and then re-initialize the source Datareplicator.

(2) Notes on a server that contains no table subject to extraction when false is specified

If you specify `false` and keep executing the HiRDB Datareplicator linkage facility without starting the source Datareplicator after its initialization, the system log file might become full even at a server that contains no table subject to extraction processing. Therefore, when you start HiRDB Datareplicator linkage, be sure to also start the source Datareplicator.

If the system log file becomes full at a server that contains no table subject to extraction processing, the action to be taken by the user depends on the specification of `pd_log_rpl_no_standby_file_opr` in the system common definition, as shown below:

- When `stop` is specified

At a server that contains no table subject to extraction processing, starting the source Datareplicator will release the system log file's extraction status. Therefore, restart the server that was terminated forcibly after the source Datareplicator starts. As a result, data linkage can continue at other servers.

- When `continue` is specified

When there is no file that can be swapped in due to the extracting status, the source HiRDB cancels HiRDB Datareplicator linkage at all back-end servers. As a result, conformity between the source and target databases is lost, and you must re-create the target database before restarting HiRDB Datareplicator linkage.

6.5.5 Checking the execution status of HiRDB Datareplicator linkage

You can use the `pdls HiRDB` command to check the execution status of HiRDB Datareplicator linkage. The `pdls` command with `rpl` specified as the type of display object displays the status of HiRDB Datareplicator linkage. This enables you determine the following about HiRDB Datareplicator linkage:

- Whether the HiRDB Datareplicator linkage facility is being used
- Status of extracting system log information from the source Datareplicator's system log file

For details about the specification of the `rpl` option in the `pdls` command, see Chapter 7. *Command Syntax*.

6.5.6 Source HiRDB handling procedure

This section explains the source HiRDB handling procedures in terms of purpose and status. The following table lists the source HiRDB handling procedures and the

paragraphs in which they are explained.

Table 6-8: Source HiRDB handling procedures

Purpose and status	Reference	
	Paragraph	Title
When initializing the source Datareplicator's environment (executing the <code>hdestart -i</code> command)	(1)	Handling procedure for re-creating the target database
When initializing the target Datareplicator's environment (executing the <code>hdsstart -i</code> command)		
When re-creating the target database		
When cancelling data linkage		
Initializing a system log file or data linkage file		
Continuing execution of data linkage	(2)	Handling procedure for continuing execution of data linkage
Modifying the definition of the database subject to extraction processing	(3)	Handling procedure for modifying the source table definition and source definition
Modifying the extraction definition		
Starting HiRDB Datareplicator linkage with the <code>pdrplstart</code> command	(4)	Handling procedure for executing the <code>pdrplstart</code> command
Cancelling HiRDB Datareplicator linkage with the <code>pdrplstop</code> command	(5)	Handling procedure for executing the <code>pdrplstop</code> command
Modifying the extraction status with the <code>plogchg</code> command	(6)	Handling procedure for executing the <code>plogchg</code> command
Re-creating a database subject to extraction processing	(7)	Handling procedure for re-creating a database subject to extraction processing
Reconstructing the system at the source HiRDB	(8)	Handling procedure for reconstructing the system at the source HiRDB
Modifying the system definition at the source HiRDB	(9)	Handling procedure for modifying the system definition at the source HiRDB
System log file in full status	(10)	Handling procedure when none of the system log files can be swapped in
Cancelling data linkage when the system log file is in extracting status and the HiRDB unit is terminated forcibly	(11)	Handling procedure for cancelling data linkage when the source HiRDB unit is terminated forcibly

Purpose and status	Reference	
	Paragraph	Title
Abnormal termination of source HiRDB	(12)	Handling procedure when the source HiRDB terminates abnormally
Adding and deleting a transmission target	(13)	Handling procedure for adding and deleting a transmission target

(1) Handling procedure for re-creating the target database

If you performed any of the operations listed below before linkage of all data had been completed, you must synchronize the data linkage environments at the source and target, initialize them, and then create the target database:

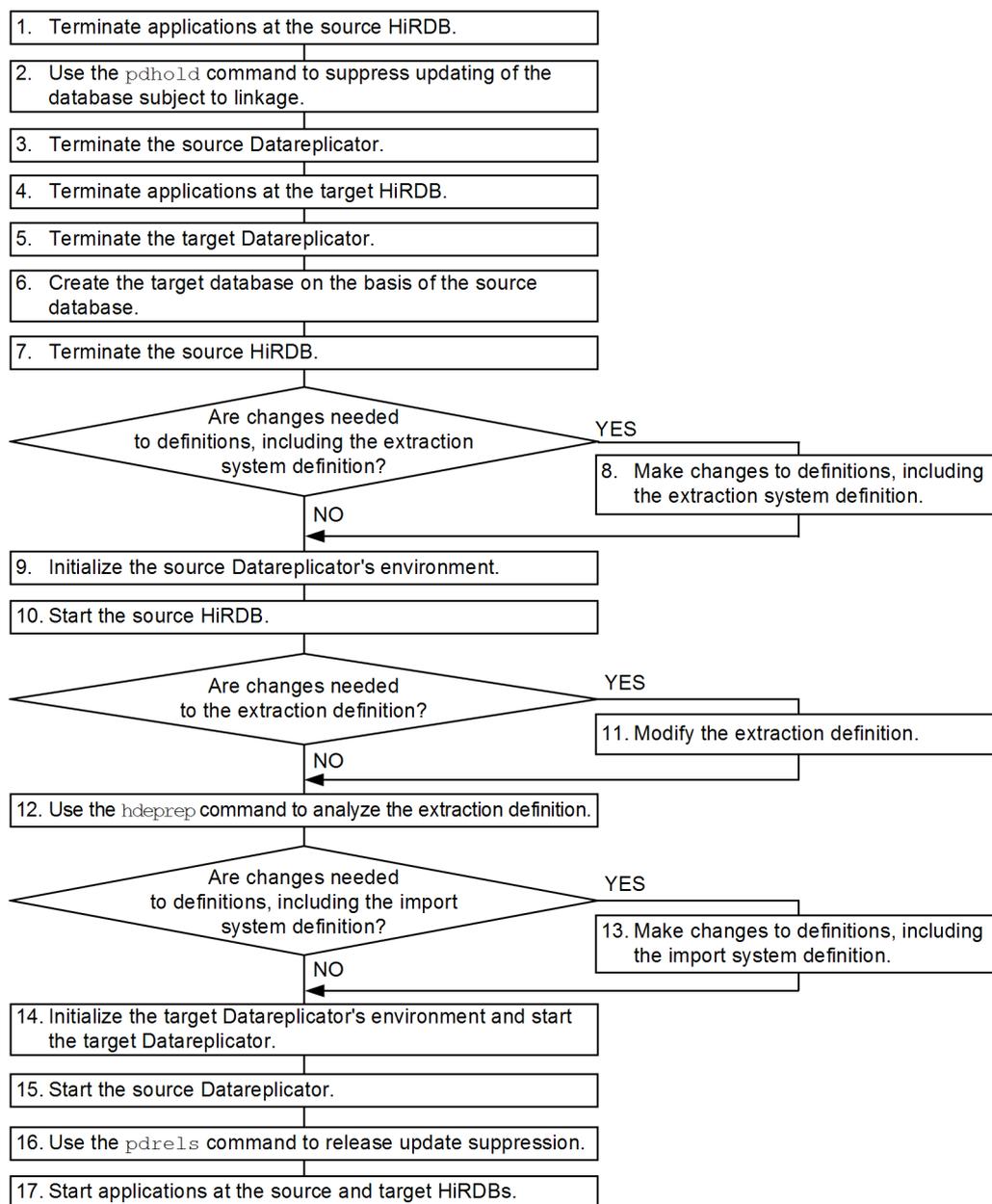
- Initializing the source Datareplicator's environment (executing the `hdestart -i` command)
- Initializing the target Datareplicator's environment (executing the `hdsstart -i` command)
- Re-creating the target database
- Stopping data linkage
- Initializing a system log file or data linkage file

For details about the initialization procedure when initialization is performed after verifying that linkage of all data has been completed and there is no need to re-create the target database, see *6.8 Changing the configuration of HiRDB and Datareplicator*.

For details about how to handle errors, see *9.2.2 Error handling methods*.

The following figure shows the handling procedure for re-creating the target database.

Figure 6-11: Handling procedure for re-creating the target database



The following explains the handling procedure steps shown in the figure; the numbers 1 to 17 below correspond to the numbers in the figure. If the source HiRDB is a parallel

server, execute the `hdeprep` and `hdestart` commands at the server where the system manager is located.

1. Terminate applications at the source HiRDB in order to maintain conformity between the source and target databases that are subject to data linkage.
2. Shut down the source database subject to data linkage with the `pdhold` command and suppress updating of the database subject to data linkage.
3. Terminate the source Datareplicator.
4. Terminate applications at the target HiRDB.
5. Terminate the target Datareplicator.
6. Create the target database on the basis of the source database. You can use the HiRDB Dataextractor to create a target database from the source database efficiently.
7. Terminate the source HiRDB.
8. If necessary, modify the following definition files:
 - Extraction system definition file
 - Transmission environment definition file
 - Extraction environment definition file
9. Initialize the source Datareplicator's environment (execute the `hdestart -i` command).
10. Specify `pd_rpl_init_start=Y` in the source HiRDB's system common definition, and then start the source HiRDB.
11. If necessary, modify the extraction definition.
12. Analyze the extraction definition with the `hdeprep` command and create an extraction definition preprocessing file.
13. If necessary, modify the following definition files:
 - Import system definition file
 - Import environment definition file
 - Import definition file
14. Initialize the target Datareplicator's environment, and then start the target Datareplicator (execute the `hdsstart -i` command).
15. Start the source Datareplicator.
16. Use the `pdrels` command to release the source database subject to data linkage from shutdown status.

17. Restart applications at the source and target HiRDBs.

(2) Handling procedure for continuing execution of data linkage

When `pd_rpl_init_start=Y` is specified in HiRDB's system common definition, HiRDB Datareplicator linkage inherits the status in which data linkage information was output to the system log at the previous source HiRDB and the system log's extraction status at the source Datareplicator regardless of whether the source HiRDB is started or restarted. Therefore, if the source Datareplicator is started with the `hdestart` command when `pd_rpl_init_start=Y` is specified, execution of data linkage continues from the previous status.

Note that if the `pdrplstop` command is executed, the database needs to be re-created even when `pd_rpl_init_start=Y` is specified because the extraction status is cancelled.

The following shows the handling procedure for continuing execution of data linkage (when `pd_rpl_init_start=Y` is specified in the system common definition):

1. Start the source HiRDB (specify `pd_rpl_init_start=Y` in the system common definition).
2. Start the source Datareplicator with the `hdestart` command.
3. Terminate the source HiRDB as required.
4. When `syncterm=true` is specified in the extraction system definition for the source Datareplicator, the source Datareplicator terminates when the source HiRDB terminates normally. If this option is not specified, execute the `hdestop` command to terminate the source Datareplicator.

(3) Handling procedure for modifying the source table definition and source definition

The following explains the handling procedure for modifying the table definition of the source database subject to data linkage and the source Datareplicator's extraction definition. You must execute the procedure described here in order to re-execute `CREATE TABLE`, even if no change has been made to the table definition of the database subject to linkage. If you modify the definitions, you must extract all of the source HiRDB's system logs at all back-end servers subject to extraction processing, and then send them to the target systems (including the update information in the extraction information queue files).

When you modify the definitions, the handling procedure depends on the circumstances, as listed below.

- `syncterm=true` is specified in the extraction system definition and the source HiRDB can be terminated normally
- `syncterm=true` is not specified in the extraction system definition or the source HiRDB cannot be terminated normally

`syncterm=true` is specified in the extraction system definition and the source HiRDB can be terminated normally

1. Terminate the source HiRDB normally. The source Datareplicator extracts all system log information from the source HiRDB, sends it to the target system, and then terminates itself automatically when transmission is completed.
2. Start the source HiRDB normally. Specify `pd_rpl_init_start=N` in the system common definition, and then execute normal start. In this case, do not execute a transaction that updates the database subject to data linkage before you start HiRDB Datareplicator linkage.
3. Modify the definitions at the source.

Table definition: Use `ALTER TABLE` and definition SQL statements to modify the definition.

Extraction definition: Modify the extraction definition.

4. If the source HiRDB is a parallel server and a table subject to extraction has been added or deleted, check and, if necessary, revise the value of `extsuppress` in the extraction environment definition.

If you have modified only within one back-end server the table definition for the database subject to linkage that is located at the source HiRDB, you can skip this step.

If you change the `extsuppress` option, initialize the source Datareplicator's environment (by executing the `hdestart -i` command) and the target Datareplicator's environment (by executing the `hdsstart -i` command).

5. Use the `hdeprep` command to analyze the modified source definitions and create the extraction definition preprocessing file. You must execute the `hdeprep` command even when changes are made only to the table definition at the source. You can execute the `hdeprep` command only while the source HiRDB is active.
6. Start the source Datareplicator with the `hdestart` command.
7. Terminate the source HiRDB normally, change the `pd_rpl_init_start` value from `N` to `Y`, and then start the source HiRDB normally.

`syncterm=true` is not specified in the extraction system definition or the source HiRDB cannot be terminated normally

1. Shut down the database subject to data linkage using the `pdhold` command to suppress database updating.
2. Have the source Datareplicator extract all system log information from the source HiRDB and send it to the target system (including all update information in the extraction information queue file).

Check the extraction and transmission status with the following commands:

- `pdls` command
 - `hdestate` command
 - Messages output by the source Datareplicator
3. Check that all system log information has been extracted and transmitted from the source HiRDB, and then terminate the source Datareplicator with the `hdestop` command.

For details about how to check for completion of system log extraction and transmission, see Step 2 in 6.8.2 *Changing the configuration of the source system*.

4. If you make either of the following changes, cancel HiRDB Datareplicator linkage at the source HiRDB with the `pdrplstop` command:
 - You add a table to a back-end server that contains no tables subject to extraction processing
 - You delete a table subject to extraction processing.
5. Modify the definitions at the source.

Table definition: Use `ALTER TABLE` and definition SQL statements to modify the definition.

Extraction definition: Modify the extraction definition.

6. If the source HiRDB is a parallel server and a table subject to extraction has been added or deleted, check and, if necessary, revise the value of `extsuppress` in the extraction environment definition.

If you have modified only within one back-end server the table definition for the database subject to linkage that is located at the source HiRDB, you can skip this step.

If you change the `extsuppress` option, initialize the source Datareplicator's environment (by executing the `hdestart -i` command) and the target Datareplicator's environment (by executing the `hdsstart -i` command).

7. Use the `hdeprep` command to analyze the modified source definitions and create the extraction definition preprocessing file. You must execute the `hdeprep` command even when changes are made only to the table definition at the source. You can execute the `hdeprep` command only while the source HiRDB is active.
8. Start the source Datareplicator with the `hdestart` command.
9. If HiRDB Datareplicator linkage is cancelled at the source HiRDB, restart it with the `pdrplstart` command.
10. Use the `pdrels` command to release the database subject to data linkage from shutdown status.

(4) Handling procedure for executing the `pdrplstart` command

The following describes the handling procedure for executing the `pdrplstart` command:

1. Use the `pdhold` command to shut down the database subject to data linkage in order to suppress database updating.
2. If the target database has not been created or conformity between the source and target databases has been lost, synchronize the data linkage environments at the source and target, initialize them, and then create the target database on the basis of the source database; for details, see (1) above.
3. Run the `pdrplstart` command at the source HiRDB to begin data linkage.
4. Start the source Datareplicator using the `hdestart` command.
5. Use the `pdrels` command to release the database subject to data linkage from shutdown status.

(5) Handling procedure for executing the `pdrplstop` command

The following describes the handling procedure for executing the `pdrplstop` command to cancel HiRDB Datareplicator linkage:

1. Cancel HiRDB Datareplicator linkage with the `pdrplstop` command.
2. Terminate the source Datareplicator with the `hdestop` command and the target Datareplicator with the `hdsstop` command. You can terminate the source and target Datareplicators at any time before initialization of data linkage.
3. Synchronize the data linkage environments at the source and target, initialize them, and then create the target database on the basis of the source database; for details, see (1) above.

(6) Handling procedure for executing the `pdlogchg` command

The following describes the handling procedure for using the `pdlogchg` command to forcibly change a system log file's extraction status. If you change the extraction status with the `pdlogchg` command while HiRDB Datareplicator linkage is executing, you need to re-create the target database.

1. Cancel HiRDB Datareplicator linkage with the `pdrplstop` command.
2. Terminate the source Datareplicator with the `hdestop` command and the target Datareplicator with the `hdsstop` command. You can terminate the source and target Datareplicators at any time before initialization of data linkage.
3. Use the `pdlogchg` command with the `-R` option specified to change the system log file from extracting status to extraction completed status. Synchronize the data linkage environments at the source and target, initialize them, and then create the target database on the basis of the source database; for details, see (1) above.

(7) Handling procedure for re-creating a database subject to extraction processing

The following describes the handling procedure for re-creating a database subject to extraction processing:

1. Cancel HiRDB Datareplicator linkage with the `pdrplstop` command.
2. Terminate the source Datareplicator with the `hdestop` command and the target Datareplicator with the `hdsstop` command. You can terminate the source and target Datareplicators at any time before initialization of data linkage.
3. Re-create the database subject to extraction processing.
4. Use the `pdhold` command to shut down the database subject to data linkage in order to suppress database updating.
5. Synchronize the data linkage environments at the source and target, initialize them, and then create the target database on the basis of the source database; for details, see (1) above.

(8) Handling procedure for reconstructing the system at the source HiRDB

The following describes the handling procedure for reconstructing the system at the source HiRDB:

1. Terminate the source HiRDB normally.
2. Terminate the source Datareplicator with the `hdestop` command and the target Datareplicator with the `hdsstop` command. You can terminate the source and target Datareplicators at any time before initialization of data linkage.
3. Reconstruct the system at the source HiRDB, specify `pd_rpl_init_start=N` in the system common definition, and then execute normal start on the source HiRDB (with the `-i` option specified).
4. Re-create the source database.
5. Synchronize the data linkage environments at the source and target, initialize them, and then create the target database on the basis of the source database; for details, see (1) above.

(9) Handling procedure for modifying the system definition at the source HiRDB

The following describes the handling procedure for modifying the system definition at the source HiRDB:

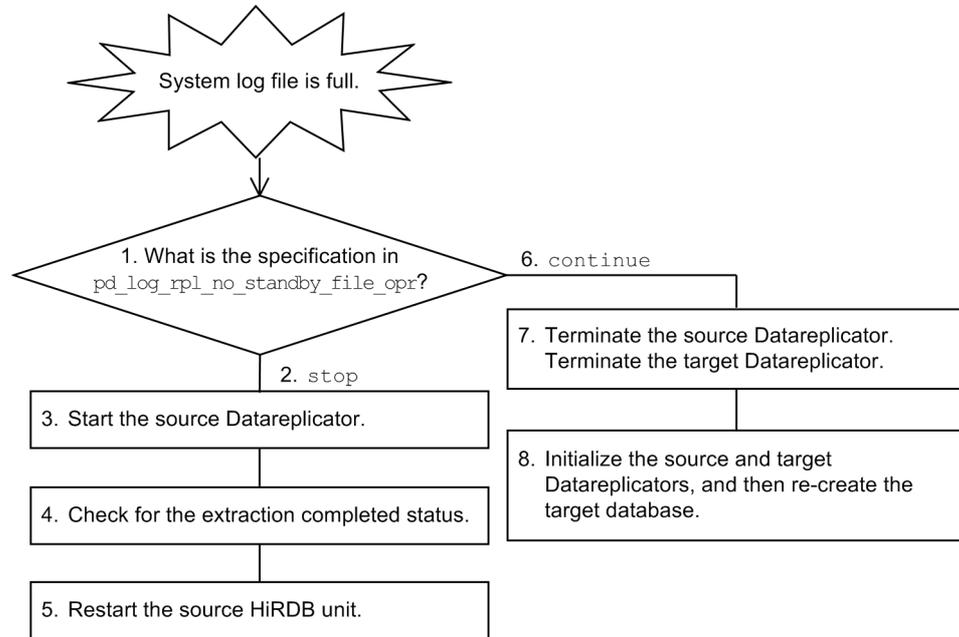
1. Terminate the source HiRDB normally.
2. Terminate the source Datareplicator with the `hdestop` command.
3. Modify the source HiRDB's system definition.
4. Start the source HiRDB.

5. Start the source Datareplicator with the `hdestart` command.

(10) Handling procedure when none of the system log files can be swapped in

The following explains the handling procedure when none of the system log files can be swapped in and there is a system log file that cannot be swapped due to its extracting status. The following figure shows the handling procedure when none of the system log files can be swapped in.

Figure 6-12: Handling procedure when none of the system log files can be swapped in



The following explains the handling procedure steps shown in the figure; the numbers 1 to 8 below correspond to the numbers in the figure:

1. The source HiRDB takes action according to the specification of `pd_log_rpl_no_standby_file_opr` in the system common definition.
2. If you have specified `stop` or omitted `pd_log_rpl_no_standby_file_opr`, the corresponding unit at the source HiRDB is terminated forcibly.
3. If the source Datareplicator is stopped, start it.
4. Check that the source Datareplicator has started using another log file (extraction from at least one file has been completed). Use the `pdlogls` command to check the extraction status of each system log, and use the `pdls -d rpl -j` command to check the progress of extraction processing.

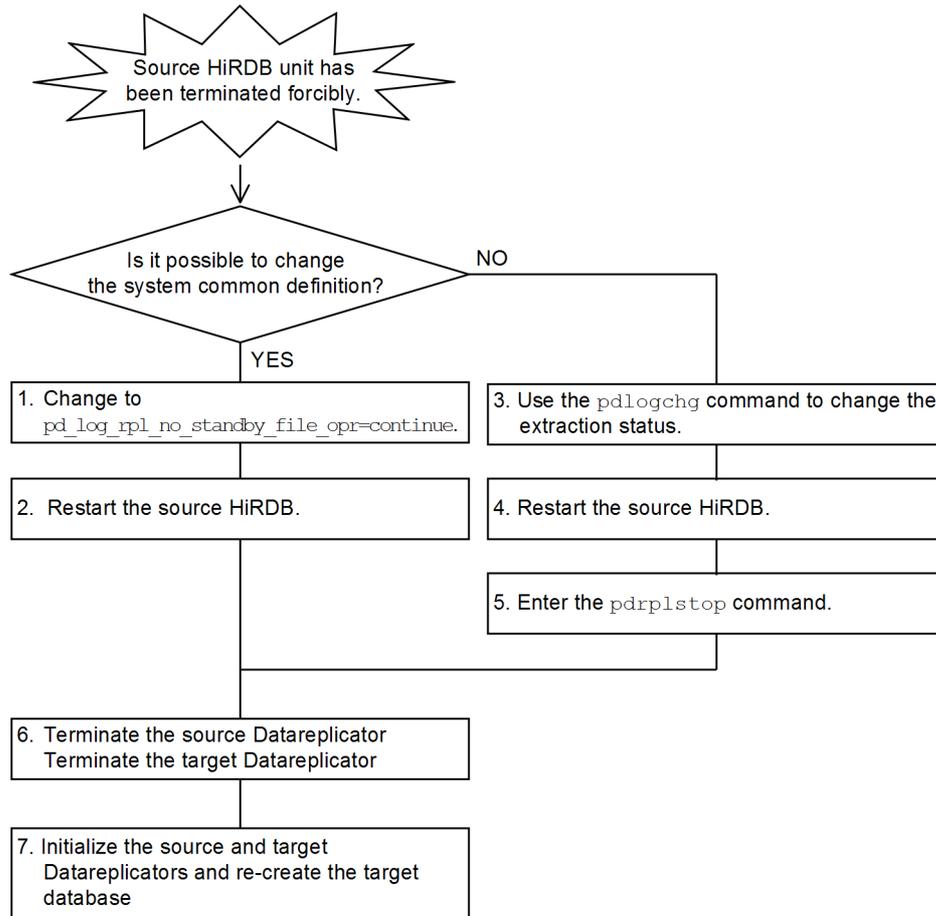
5. Restart the unit at the source HiRDB.
6. If you have specified `continue`, cancel HiRDB Datareplicator linkage and resume processing only at the source HiRDB. When there are no more files that can be swapped in due to the extracting status, the source HiRDB cancels HiRDB Datareplicator linkage and continues operation only at the source HiRDB. When there are no more files that can be swapped in, the source HiRDB releases the extraction status of all system log files.
7. Terminate the source Datareplicator with the `hdestop` command and the target Datareplicator with the `hdsstop` command. You can terminate the source and target Datareplicators at any time before initialization of data linkage.
8. Synchronize the data linkage environments at the source and target, initialize them, and then create the target database on the basis of the source database; for details, see (1) above.

(11) Handling procedure for cancelling data linkage when the source HiRDB unit is terminated forcibly

This subsection describes the handling procedure for cancelling data linkage and continuing processing only at the source HiRDB when the source HiRDB unit is terminated forcibly according to HiRDB's system common definition because the system log file is in extracting status and the problem cannot be resolved by starting the source Datareplicator.

The following figure shows the handling procedure for cancelling data linkage when the source HiRDB unit is terminated forcibly.

Figure 6-13: Handling procedure for cancelling data linkage when the source HiRDB unit is terminated forcibly



The following explains the handling procedure steps shown in the flowchart in Figure 6-9; the numbers 1 to 7 below correspond to the numbers in the figure:

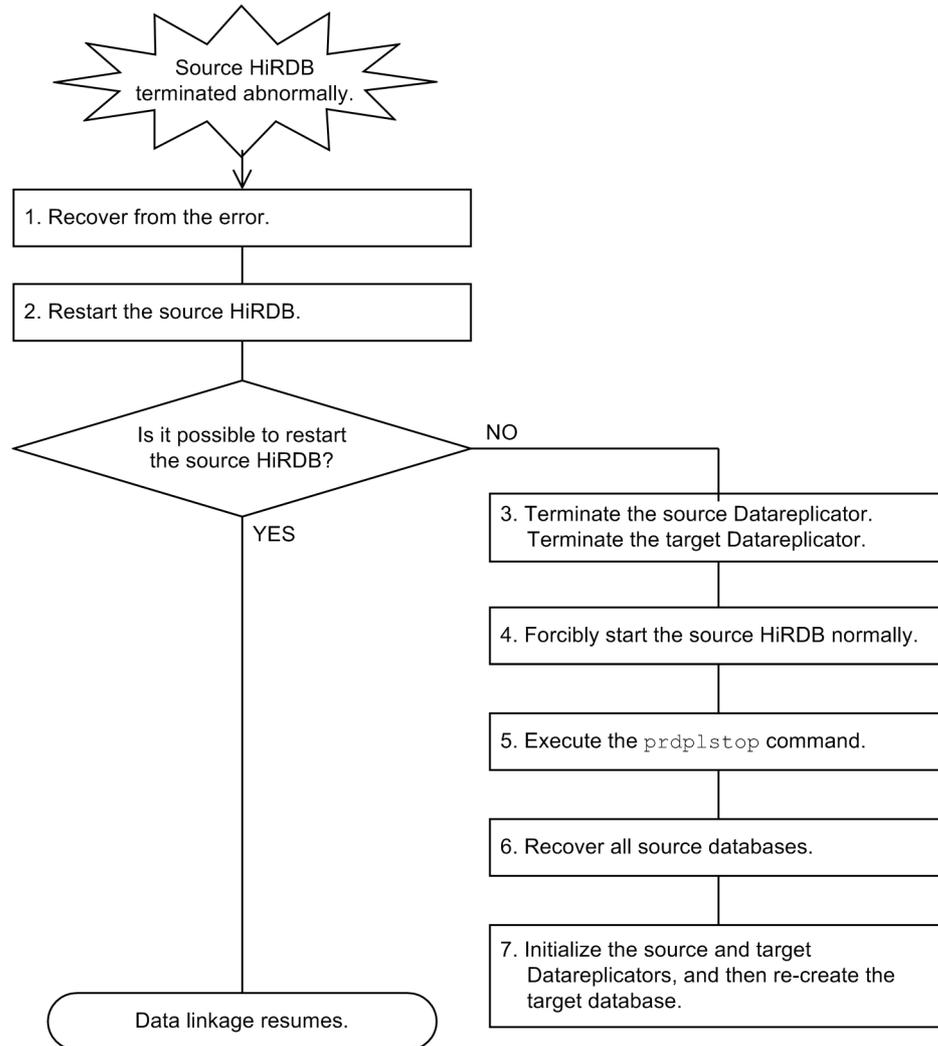
1. If possible, change the `pd_log_rpl_no_standby_file_opr` operand specification to `continue` in the system common definition.
2. Restart the source HiRDB.
3. If it is not possible to change the definition, execute the `pdlogchg` command with the `-R` option specified on all log files to forcibly change the status from extracting to extraction completed.
4. Restart the source HiRDB.

5. Use the `pdrplstop` command to cancel HiRDB Datareplicator linkage.
6. Terminate the source Datareplicator with the `hdestop` command and the target Datareplicator with the `hdsstop` command. You can terminate the source and target Datareplicators at any time before initialization of data linkage.
7. Synchronize the data linkage environments at the source and target, initialize them, and then create the target database on the basis of the source database; for details, see (1) above.

(12) Handling procedure when the source HiRDB terminates abnormally

The following figure shows the handling procedure when the source HiRDB terminates abnormally.

Figure 6-14: Handling procedure when the source HiRDB terminates abnormally



The following explains the handling procedure steps shown in the flowchart in Figure 6-10; the numbers 1 to 7 below correspond to the numbers in the figure:

1. Eliminate the cause of the abnormal termination of the source HiRDB.
2. Restart the source HiRDB.
3. Terminate the source Datareplicator with the `hdestop` command and the target Datareplicator with the `hdsstop` command. You can terminate the source and

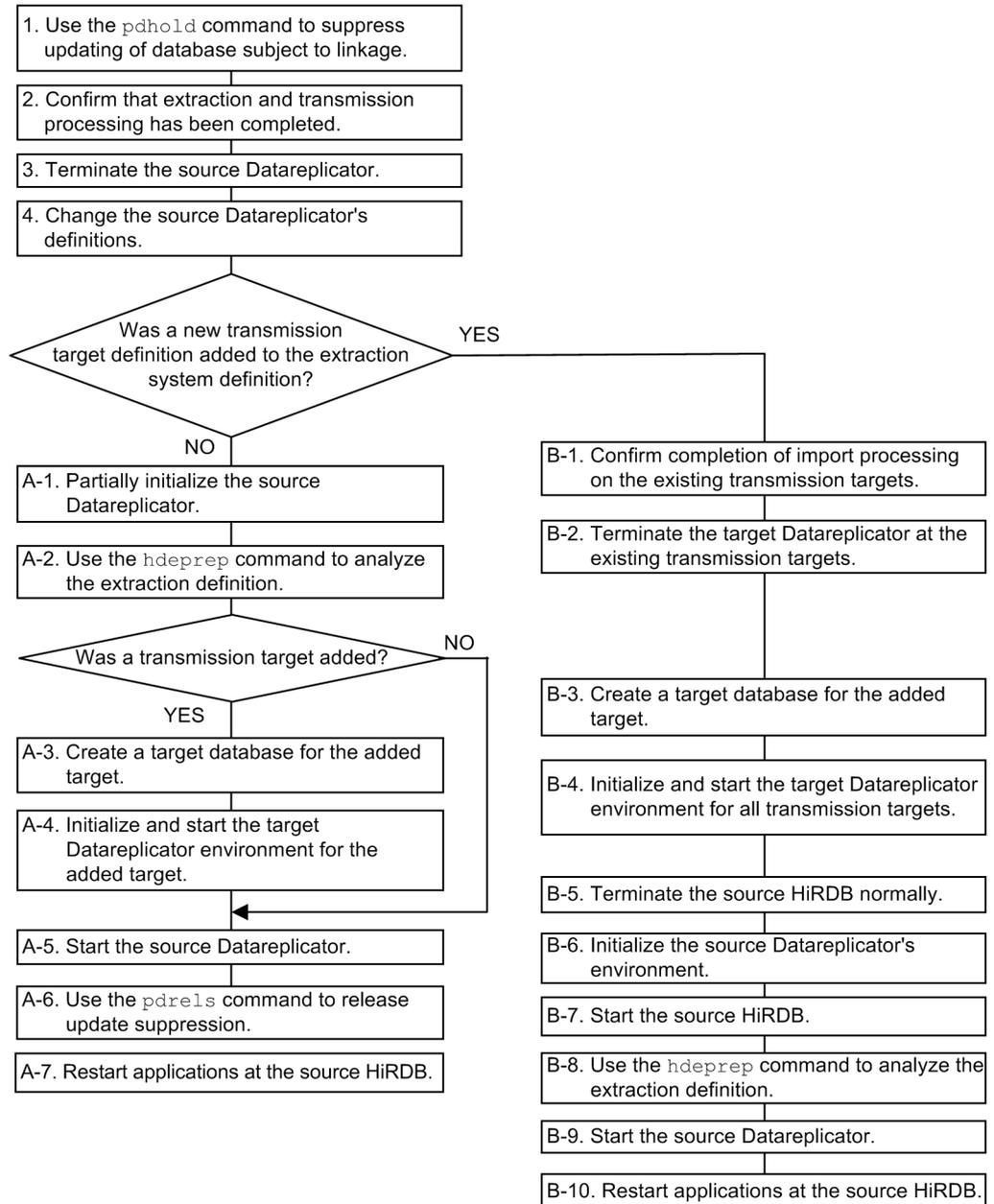
target Datareplicators at any time before initialization of data linkage.

4. Forcibly start the source HiRDB normally.
5. Use the `pdrplstop` command to cancel HiRDB Datareplicator linkage.
6. Re-create the source database.
7. Synchronize the data linkage environments at the source and target, initialize them, and then create the target database on the basis of the source database; for details, see (1) above.

(13) Handling procedure for adding and deleting a transmission target

The following figure shows the handling procedure for adding and deleting a transmission target to and from the source Datareplicator.

Figure 6-15: Handling procedure for adding and deleting a transmission target



The following explains the handling procedure steps shown in the figure. Numbers 1 through 4 correspond to the numbers in the figure.

1. Use the `pdhold` command to shut down the database subject to data linkage and suppress updating of the database.
2. Extract all of the source HiRDB's system logs at the source Datareplicator and make sure that all necessary data has been sent to the target (including the update information in the extraction information queue files).

To check the extraction and transmission status, use the following methods:

- `pdls` command
 - `hdestate` command
 - Messages issued by the source Datareplicator
3. Make sure that all of the source HiRDB's system logs have been extracted and transmitted, and then terminate the source Datareplicator with the `hdestop` command.

For details about how to check for completion of system log extraction and transmission, see Step 2 in *6.8.2 Changing the configuration of the source system*.

4. Change the definitions at the source.

When adding a transmission target:

If the extraction system definition contains a missing number for a transmission target definition, change the corresponding definition to the transmission target to be added. If there is no such missing number, add a definition for the new transmission target (`sendid` and `senddef` operands).

When deleting a transmission target:

Change the applicable transmission target definition to a missing number in the extraction system definition.

The subsequent portion of the procedure depends on whether a new transmission target was added to the extraction system definition in step 4 (whether new `sendid` and `senddef` operands were added to the extraction system definition).

If you did not add a new transmission target, follow procedure A; if you added a new transmission target, follow procedure B.

Procedure A

1. Execute partial initialization on the source Datareplicator for the transmission target whose definition was changed.
2. Execute the `hdeprep` command to create an extraction definition preprocess.
3. Create a target database for the new transmission target on the basis of the source database.

4. Initialize the environment of the target Datareplicator at the new transmission target.
Before initializing the target Datareplicator environment, you must create the following definition files:
 - Import system definition file
 - Import environment definition file
 - Import definition file
5. Execute the `hdestart` command to start the source Datareplicator.
6. Execute the `pdrels` command to release the source database subject to data linkage from shutdown status.
7. Restart applications at the source HiRDB.

Procedure B

1. Make sure that all data sent from the source has been imported into the target Datareplicators at all existing transmission targets.
Execute the `hdsstate` command to verify that the update information in the import information queue files has been imported. All update information has been imported when `write position in COMMUNICATION INFORMATION` and `read position in REFLECTION INFORMATION` are the same in the status information obtained by executing the `hdsstate` command.
2. Terminate the target Datareplicators at all existing transmission targets with the `hdsstop` command.
3. Create a target database for the new transmission target on the basis of the source database.
4. Initialize the environment of the target Datareplicators at all existing transmission targets as well as the new transmission target.
Before initializing the environment of the target Datareplicator at the new transmission target, you must create the following definition files:
 - Import system definition file
 - Import environment definition file
 - Import definition file
5. Use the `pdstop` command to terminate the source HiRDB normally.
6. Execute the `hdsstart -i` command to initialize the source Datareplicator's environment.

7. Execute the `pdstart` command to start the source HiRDB.
8. Execute the `hdeprep` command to create an extraction definition preprocess.
9. Execute the `hdestart` command to start the source Datareplicator.
10. Restart applications at the source HiRDB.

6.5.7 Notes on handling when `syncterm=true` is specified

The following notes on Datareplicator operation apply when `syncterm=true` is specified in the extraction system definition.

(1) *Terminating Datareplicator before synchronized termination with HiRDB*

When you specify `syncterm=true`, Datareplicator continues processing up to the synchronization point (detection of HiRDB normal termination log) and then terminates itself automatically. If the source HiRDB is a parallel server and is terminated by a command issued by the user or due to an error before Datareplicator reaches a synchronization point, some servers might have reached the synchronization point, but some might not. If you restart data linkage by specifying `syncterm=true` in such a case, extraction processing might be terminated before the most recent synchronization point is reached because the synchronization point identified by each server is different. When you specify `syncterm=true`, you must be sure to execute Datareplicator up to the synchronization point (restart only Datareplicator and allow all servers to reach the most recent synchronization point).

(2) *Restarting Datareplicator after synchronous termination*

When you specify `syncterm=true` and Datareplicator terminates synchronously with the HiRDB, you must start HiRDB and start HiRDB Datareplicator linkage before restarting Datareplicator with the `syncterm=true` specification. If restarted while data linkage is inactive at the HiRDB, Datareplicator will terminate itself automatically.

(3) *Operation when communication has not been established with the target Datareplicator*

When you specify `syncterm=true` but communication has not been established with the target Datareplicator, transmission processing will not stop because it cannot send update information in the extraction information queue file to the target Datareplicator even if the HiRDB terminates normally. In such a case, when the target Datareplicator is started and communication is established, the transmission process stops automatically when it finishes sending all update information.

6.5.8 Notes on handling when `sendcontrol=sendmst` is specified

The following notes on the operating Datareplicator apply when `sendcontrol=sendmst` is specified in the extraction system definition.

(1) Target Datareplicator connection/disconnection timing

When you specify `sendcontrol=sendmst`, communication with the target Datareplicator is established and broken at the specified transmission interval. However, if there is no update information to be sent when the transmission interval is reached, connection or disconnection processing does not occur. Therefore, when `sendcontrol=sendmst` is specified, the following operation is different from when `sendcontrol=nodemst` is specified:

- The target Datareplicator does not detect the communication line status between transmission intervals (while communication with the target Datareplicator is off).
- If you execute the normal termination command (`hdsstop`) on the target Datareplicator before sending any update information to it after the transmission processing start request, the target Datareplicator terminates itself without waiting for termination of the source Datareplicator.

(2) Suppression of transmission process reutilization

If an error occurs in a transmission process due to a memory shortage while transmission is underway, the source Datareplicator cancels transmission processing for the corresponding destination and suppresses reutilization of this transmission process. To reuse this transmission process, you must restart the source Datareplicator.

6.6 Startup and termination of the target Datareplicator

The following figure shows the procedures for starting and terminating the target Datareplicator.

Figure 6-16: Procedures for starting and terminating the target Datareplicator



6.6.1 Starting the target Datareplicator

This subsection explains the target Datareplicator's start method and start modes.

For details about the import processing start method, see *4.7.4 Designing the import processing start method*.

(1) How to start the target Datareplicator

When you execute the `hdsstart` command at the target system, the target Datareplicator starts in accordance with the import system definition and the import environment definition. In the Windows edition of Datareplicator, execute the command at the command prompt.

Before you use the `hdsstart` command, specify in the import system definition file a password that will be required to establish connection with the import HiRDB. Otherwise, it will be possible for anyone to connect to the target HiRDB without entering a password.

To initialize the target Datareplicator's environment, execute the `hdsstart -i` command. To initialize the environment only for a specific source, use the `hdsstart -i -D` command to initialize the environment only for the corresponding source.

For details, see the *hdsstart* command in Chapter 7. *Command Syntax*.

In the Windows edition of Datareplicator, you can also use either of the following methods to start the target Datareplicator:

- Start the import service manually
- Set the import service to start automatically when Windows starts

To start the import service manually:

1. In **Control Panel**, double-click the **Services** icon.
2. Under **Service**, choose **HiRDB Datareplicator (Target Site)**.
3. In the **Startup Parameters** field, specify the startup options (options for the *hdsstart* command).
4. Click the **Start** button.

To set the import service to start automatically when Windows starts:

The procedure for setting the import service to start automatically when Windows starts is explained below. In this case, you cannot specify any start options (options in the *hdsstart* command).

1. In **Control Panel**, double-click the **Services** icon.
2. Under **Service**, choose **HiRDB Datareplicator (Target Site)**.
3. Click the **Startup** button.
4. In the Service dialog box, under **Startup Type**, choose **Automatic**.
5. Click the **OK** button.
6. Click the **Close** button.

The next time you start Windows, the import service will start automatically.

(2) Start modes for the target Datareplicator

There are four start modes for the target Datareplicator.

For details about the start modes, see the *hdsstart* command in Chapter 7. *Command Syntax*.

Initial start

The initial start mode starts the target Datareplicator in accordance with the import system definition without inheriting the previous session. In this mode, the target Datareplicator initializes the following files:

- Import information queue files
- Import status files

- Import master status file
- Unimported information files
- Import error information files
- Import trace files

Partial initial start

The partial initial start mode initializes the import environments for specified data linkage identifiers only, in accordance with the corresponding import system definition (partial initialization) and starts the target Datareplicator's processing without inheriting the settings of the previous session. In this mode, the target Datareplicator initializes the following files for each specified data linkage identifier:

- Import information queue files
- Import status files
- Unimported information files

The target Datareplicator's processing for any data linkage identifier that is not specified is the same as in the normal start or restart mode.

Normal start

The normal start mode starts the target Datareplicator in accordance with the import system definition without inheriting the settings of the previous session.

Restart

When import processing in the previous session terminated with an error, the restart mode starts the target Datareplicator with the same settings that were in effect for the previous session in order to guarantee conformity between source and target databases.

6.6.2 Terminate the target Datareplicator

The following explains the termination procedure and termination modes for the target Datareplicator.

For details about how to stop import processing, see *4.7.5 Designing the import processing stop method*.

(1) How to terminate the target Datareplicator

If you execute the `hdsstop` command after the source system has terminated, the target Datareplicator terminates according to the import environment definition. In the Windows edition of Datareplicator, execute the command at the command prompt.

To terminate the target Datareplicator, make sure that you use the `hdsstop` command. If you terminate the target Datareplicator by using an OS command to kill the process, the results might not be as expected.

In the Windows edition of Datareplicator, you can also use the following method to terminate the target Datareplicator (immediate termination).

Terminate the import service manually:

1. In **Control Panel**, double-click the **Services** icon.
2. Under **Service**, choose **HiRDB Datareplicator (Target Site)**.
3. Click the **Stop** button.

If you terminate Windows while the target Datareplicator is running, the target Datareplicator is terminated forcibly. Once the import service is terminated, executing a forced termination command before the import service actually stops will not result in forced termination.

(2) Termination modes for the target Datareplicator

There are four termination modes for the target Datareplicator.

For details about the termination modes, see the *hdsstop* command in Chapter 7. *Command Syntax*.

Normal termination

After executing the `hdsstop` command, the target Datareplicator terminates when import processing is completed through the source system's activity unit. If the source database is a HiRDB, the activity unit is from normal startup to normal termination of the entire source system. For details about the activity unit when the source database is a mainframe database, see the *VOS3 XDM/DS* manual.

Event termination

After executing the `hdsstop` command, the target Datareplicator terminates when import processing detects the first event sent from the source system. If the target Datareplicator completes processing through the source system's activity unit before detecting the event, it terminates immediately without waiting for the event.

Immediate termination

After executing the `hdsstop` command, the target Datareplicator terminates when it finishes importing the current update information being processed.

Forced termination

After executing the `hdsstop` command, the target Datareplicator terminates forcibly regardless of the import status of the update information. If synchronization point processing is underway, the target Datareplicator terminates upon completion of the synchronization point processing.

Executing forced termination on the target Datareplicator might result in a loss of conformity between the source and target databases. If you use the restart mode the next time you start the target Datareplicator, the target database will be restored

automatically.

6.7 Operation of the target Datareplicator

This section explains the target Datareplicator operating procedures.

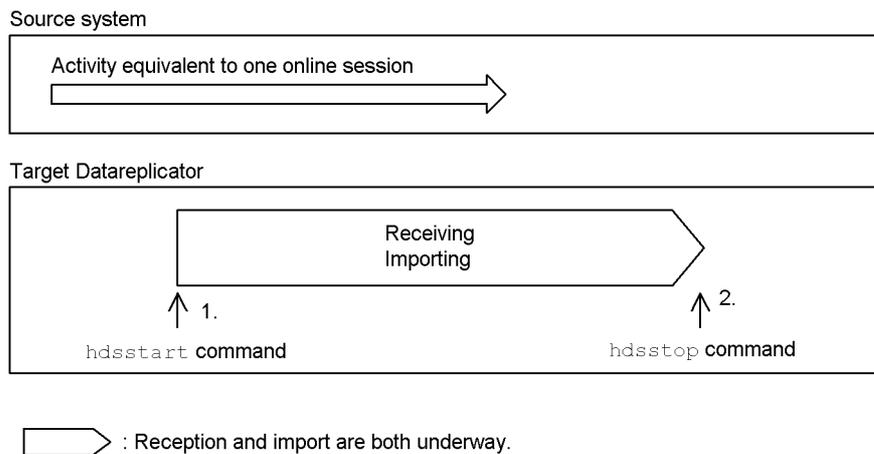
6.7.1 Handling of import processing

This section explains the import processing handling procedures.

(1) Starting the target Datareplicator using the simultaneous start method

The simultaneous start method starts import processing simultaneously with startup of the target Datareplicator. This method enables you to import update information in the order it is received. Figure 6-17 shows an example of target Datareplicator operation using the simultaneous start method, and Table 6-9 explains the handling procedure and target Datareplicator processing when the simultaneous start method is used.

Figure 6-17: Example of target Datareplicator operation using the simultaneous start method



Note: The numbers in the figure (1 and 2) correspond to the numbers in Table 6-9.

Table 6-9: Handling procedure and target Datareplicator processing (simultaneous start)

No.	Handling procedure	Target Datareplicator processing
1	Execute the <code>hdsstart</code> command (for an initial startup, execute <code>hdsstart -i</code> command).	The target Datareplicator starts and begins receiving and importing update information sent from the source system.

No.	Handling procedure	Target Datareplicator processing
2	Execute the <code>hdsstop</code> command.	The target Datareplicator terminates after importing all update information for the source system's online application (equivalent to one session).

Definition example

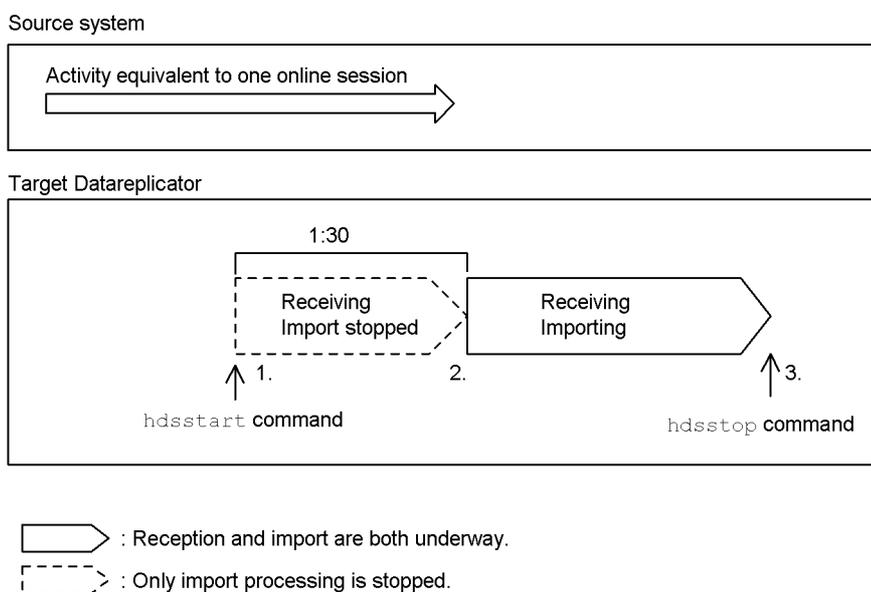
The following is an example of the import environment definition for starting the target Datareplicator with the simultaneous start method:

```
set startmode=trn
```

(2) Starting the target Datareplicator using the delayed (time-specified) start method (time-specified start method)

This method starts import processing after a user-specified amount of time elapses since startup of the target Datareplicator. Figure 6-18 shows an example of target Datareplicator operation using the delayed start (time-specified start) method, and Table 6-10 explains the handling procedure and target Datareplicator processing when the delayed start (time-specified start) method is used.

Figure 6-18: Example of target Datareplicator operation using the delayed start (time-specified start) method



Note: The numbers in the figure (1 through 3) correspond to the numbers in Table 6-10.

Table 6-10: Handling procedure and target Datareplicator processing (delayed start (time-specified start) method)

No.	Handling procedure	Target Datareplicator processing
1	Execute the <code>hdsstart</code> command.	The target Datareplicator starts and begins receiving update information sent from the source system.
2	--	The target Datareplicator automatically starts import processing 1 hour and 30 minutes after its startup.
3	Execute the <code>hdsstop</code> command.	The target Datareplicator terminates after importing all update information for the source system's online application (equivalent to one session).

Legend:

--: No action is needed.

Definition example

The following is an example of the import environment definition for starting the target Datareplicator with the delayed start (time-specified start) method, where the target Datareplicator starts import processing using the table-based import method 1 hour and 30 minutes after its startup:

```
set startmode=spd
set breaktime=01:30
set breakmode=tbl
```

(3) Starting the target Datareplicator using the delayed (command) start method (command start method)

This method starts the import processing by executing a command in the source system.

To start the target Datareplicator by using the delayed start method (command start method):

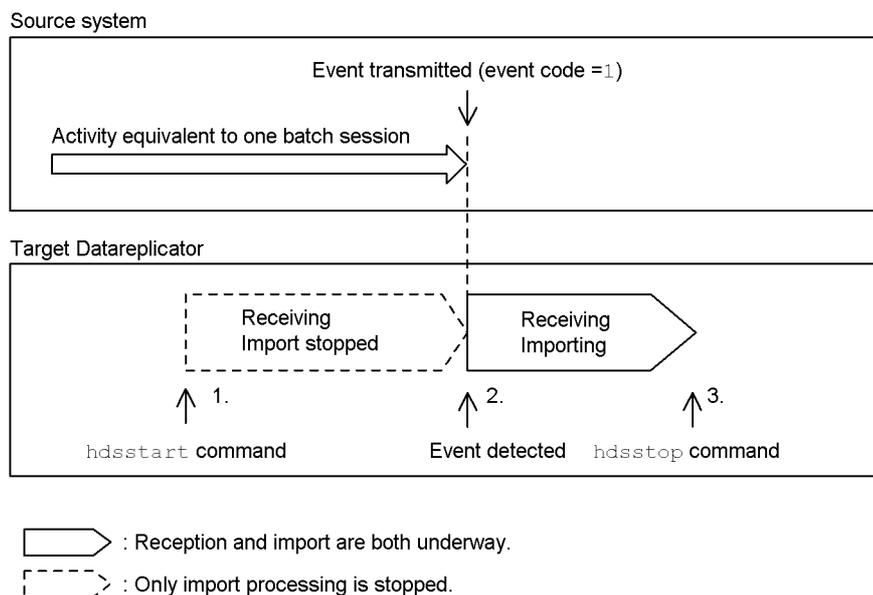
1. Specify `spd` in the `startmode` operand in the import environment definition.
2. Execute the `hdsstart` command at the target system to start the target Datareplicator. Only reception processing begins according to the import environment definition.
3. Execute the `hdsrfctl -m trn` or `hdsrfctl -m tbl` command at the target system. Import processing begins using the specified import method.

(4) Starting the target Datareplicator using the delayed (event) start method (event start method)

This method starts import processing when an event from the source system is

detected. Figure 6-19 shows an example of target Datareplicator operation using the delayed start (event start) method, and Table 6-11 explains the handling procedure and target Datareplicator processing when the delayed start (event start) method is used.

Figure 6-19: Example of target Datareplicator operation using the delayed start (event start) method



Note: The numbers in the figure (1 through 3) correspond to the numbers in Table 6-11.

Table 6-11: Handling procedure and target Datareplicator processing (delayed start (event start) method)

No.	Handling procedure	Target Datareplicator processing
1	Execute the <code>hdsstart</code> command.	The target Datareplicator starts and begins receiving update information sent from the source system.
2	--	The target Datareplicator starts import processing when it detects an event corresponding to the import restart event.
3	Execute the <code>hdsstop</code> command.	The target Datareplicator terminates after importing all update information for the source system's batch application (equivalent to one session).

Legend:

--: No action is needed.

Definition example

The following is an example of the import environment definition for starting the target Datareplicator with the delayed start (event start) method, where the target Datareplicator starts import processing using the transaction-based import method after receiving an event with event code 1:

```
set startmode=spd
set eventretrn=1
```

(5) Terminating import processing

To terminate import processing while keeping reception processing active:

1. Execute the `hdsrfctl -m spd` command at the target system.
2. Generate an event at the source system. Import processing stops when the target system detects the event. For details about how to issue events at the source system when the source database is a HiRDB, see the `hdeevent` command (issue an event at the source Datareplicator) in 7. *Command Syntax*. For details about how to issue events at the source system when the source database is a mainframe database, see the *VOS3 XDM/DS* manual.

Alternatively, execute the `hdsrfctl -m immediate` command at the target system.

(6) Restarting import processing by specifying an import processing method

If import processing has stopped due to an import definition error or an SQL error, or if only reception processing was activated during startup, you can restart import processing only by specifying the desired import processing method.

To restart import processing by specifying an import processing method:

1. Execute the `hdsrfctl -m trn` or `hdsrfctl -m tbl` command at the target system.
2. At the source system, generate an event corresponding to the `eventretrn` or `eventretbl` operand in the import environment definition. When the target system detects the event, import processing is restarted by means of the specified method.

(7) Switching the import method during operation

You can stop import processing while data linkage processing is underway in accordance with the import environment definition, and then restart processing at a desired time using a different import method (transaction-based import method or table-based import method).

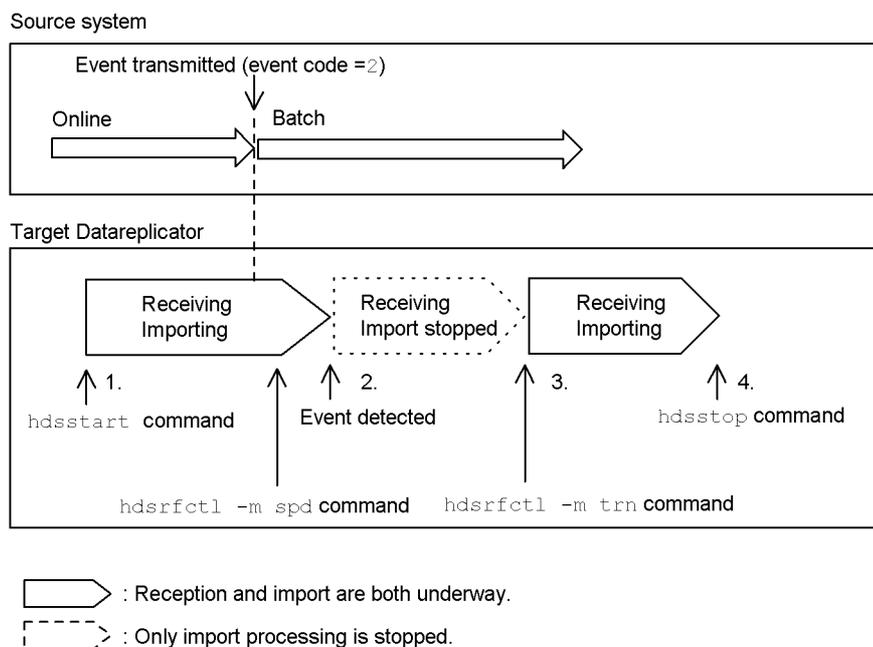
To switch the import method during operation:

1. Execute the `hdsrfctl -m spd` command at the target system to stop import processing.

2. Execute the `hdsrfctl -m trn` or `hdsrfctl -m tbl` command at the target system.
3. At the source system, generate an event that corresponds to the `eventtrn` or `eventtbl` operand in the import environment definition. The import method is switched when the target system detects the event generated at the source system.

Figure 6-20 shows an example of operation involving switching of the import method, and Table 6-12 explains the handling procedure and target Datareplicator processing when the import method is switched.

Figure 6-20: Example of operation to switch the import method



Note: The numbers in the figure (1 through 4) correspond to the numbers in Table 6-12.

Table 6-12: Handling procedure and target Datareplicator processing (switching the import method)

No.	Handling procedure	Target Datareplicator processing
1	Execute the <code>hdsstart</code> command.	The target Datareplicator starts and begins receiving and importing update information sent from the source system (online application).

No.	Handling procedure	Target Datareplicator processing
2	Execute the <code>hdsrfctl -m spd</code> command before terminating import of update information for the online application (in accordance with the import environment definition, the target Datareplicator is to change to the table-based import method to resume import processing when it finishes importing update information for the online application).	The target Datareplicator terminates only the import processing after detecting an event (event code = 2). (If executed, the <code>hdsrfctl</code> command takes precedence over the specification of <code>eventtbl=2</code> in the import environment definition.)
3	Execute the <code>hdsrfctl -m trn</code> command before terminating reception of update information for the batch application.	When the command is executed, the target Datareplicator restarts import processing on the update information for the batch application.
4	Execute the <code>hdsstop</code> command.	The target Datareplicator terminates after terminating import processing on the update information for the batch application.

Definition example

Shown below is an example of the import environment definition for switching the import method during operation. In this example, the target Datareplicator ignores the specification of `eventtbl` in the import environment definition because the `hdsrfctl` command is used to control the import processing.

```
set eventtbl=2
```

(8) Changing the import processing method during operation

The target Datareplicator permits you to change the import processing method during operation.

To change the import processing method during operation:

1. Execute the `hdsrfctl -m spd` command at the target system in order to stop import processing.
2. Execute the `hdsrfctl -m trn` or `hdsrfctl -m tbl` command at the target system.
3. At the source system, generate an event corresponding to the `eventtrn` or `eventtbl` operand in the import environment definition. When the target system detects the event, the import processing method changes.

(9) Restarting some import groups in error stop status

The target Datareplicator cannot restart selected import groups. If some import groups

have terminated due to an SQL error while import processing is underway with the table-based import method, follow the procedure below.

To restart some import groups in error stop status:

1. Execute the `hdsrfctl -m immediate` or `hdsstop -t immediate` command at the target system to terminate import processing on the current import groups.
2. Eliminate the cause of the error and execute the `hdsrfctl -m tbl` or `hdsstart` command at the target system. The entire import processing starts, with the previous mode inherited.

(10) Establishing data linkage from a source database using SAM files

The source system uses a file transfer program to send a SAM file containing update information to the target system. This SAM file is stored in the target Datareplicator directory specified by the source system's file transfer program.

The target Datareplicator user executes the `hdssamqin` command to store the update information from the SAM file into the import information queue file. If active, the target system's import processing reads the update information from the import information queue file and updates the target database.

6.7.2 Handling of the files used with the target Datareplicator

This section explains the procedures for handling the files used with the target Datareplicator.

For details about the contents of the files, see *3.3.2 Files and processes used during import processing*.

For details about the preparation of the files, see *4.7.2 Preparation of the files used with the target Datareplicator*.

Note: Do not replace files, regardless of whether Datareplicator is running or stopped. If any of these files is replaced, Datareplicator might malfunction.

(1) Import system definition file handling

For details about how to handle the import system definition file, see *5.8.2 Modifying defined information*.

(2) Import environment definition file handling

For details about how to handle the import environment definition file, see *5.9.2 Modifying defined information*.

(3) Import definition file handling

For details about how to handle the import definition file, see *5.10.3 Modifying defined information*.

(4) Import information queue file handling method

The following explains how to handle the import information queue files.

Modifying a filename, the file size, or the number of files

When using a UNIX regular file or Windows file:

1. Terminate the source system.
2. Terminate the target Datareplicator normally.
3. Use a text editor to modify the corresponding operands in the import environment definition.
4. Execute initial start or partial initial start on the target Datareplicator.
5. Start the source system.

When using a UNIX character special file:

1. Terminate the source system.
2. Terminate the target Datareplicator normally.
3. Use an OS command to re-create the import information queue files in character special file format.
4. Use a text editor to modify the corresponding operands in the import environment definition as appropriate for the created import information queue files.
5. Execute initial start or partial initial start on the target Datareplicator.
6. Start the source system.

(5) Import status file handling

The following explains how to handle an import status file.

Changing the filename or file size

When using a UNIX regular file or Windows file:

1. Terminate the source system.
2. Terminate the target Datareplicator normally.
3. Use a text editor to modify the corresponding operands in the import environment definition.
4. Execute initial start or partial initial start on the target Datareplicator.
5. Start the source system.

When using a UNIX character special file:

1. Terminate the source system.

2. Terminate the target Datareplicator normally.
3. Use an OS command to delete the previous import status file.
4. Use an OS command to re-create the import status file in character special format.
5. Use a text editor to modify the corresponding operands in the import environment definition as appropriate for the created import status file.
6. Execute initial start on the source Datareplicator.
7. Start the source system.

(6) Import master status file handling

The following explains how to handle the import master status file.

The import master status file is used to store execution results during initial start. It is created automatically when Datareplicator starts.

(7) Import error information file handling

The following explains how to handle the import error information files.

(a) Changing the maximum size

To change the maximum size of the import error information files:

1. Terminate the source system.
2. Terminate the target Datareplicator.
3. Use a text editor to modify the appropriate operand in the import system definition.
4. Execute normal start on the target Datareplicator.
5. Start the source system.

(b) Saving an import error information file

When an import error information file subject to data accumulation becomes full, file swapping occurs. Datareplicator re-creates the next import error information file so that it can be swapped in.

To save the contents of an import error information file before it is re-created:

1. Use an OS command to check the most recent update dates and times of the import error information files.
2. Use an OS command to copy the import error information file with the older update date and time into a new file under any name. Datareplicator issues a message (KFRB00051-I or KFRB00052-I) whenever import error information files are to be swapped or whenever an import error information file is to be closed during Datareplicator operation; when this message is issued, make a backup

copy if necessary.

(c) Outputting import error information to the standard output

When the import error information file being used for data accumulation becomes full, file swapping occurs. Datareplicator re-creates the next import error information file that is to be swapped in.

To output the contents of the import error information file to the standard output before the file is re-created:

1. Use an OS command to check the most recent update dates and times of the import error information files.
2. Use an OS command to output the import error information file with the older update date and time to the standard output.

(d) Example of import error information file output

The following figure shows an example of the output from an import error information file.

Figure 6-21: Example of the output from an import error information file

```

*****
1.  Fri Mar 29 14:33:14 2002  process: hdsreflect[othergrp] (10714)
    function: hds_rfc_comctl
2.  Errorcode: KFRB03009-I 99 c2 (hdsreflect[othergrp]) Import process completed
    synchronization point processing. Reason code = END OF CONNECTION, file information =
    [0, 5120]
3.  information: SQL count = [Insert:4(4), Update:5(5), Delete:4(4), Purge:1(2),
    timestamp:0(0), Commit:1(2)]
    *****
    Fri Mar 29 14:33:14 2002  process: hdsreflect[othergrp] (10714)
    function: hds_rfc_main
    errorcode: KFRB03006-I 99 c2 (hdsreflect[othergrp]) Import process detected an event.
    Event code = -1
    information:
    *****

```

Explanation:

1. `Fri ... 2002, process ..., function ...`
`Fri ... 2002`: Day of week, month, date, time (*hh:mm:ss*), and year when error occurred
`process`: Name and number of internal process resulting in the error
`function`: Name of internal function resulting in the error
2. `errorcode`: Error code of the error
3. `information`: Detailed information about the error

(e) Output destinations other than files

Datareplicator outputs information to the import error information files so that it can be used in the event of errors or to achieve automatic operation.

In the case of UNIX Datareplicator, this information is also output to the syslog file. To output information to the syslog file, you must specify `true` in the `syslogout` operand in the import system definition. The following table shows the output destinations for the contents of an import error information file.

Table 6-13: Output destinations of the contents of an import error information file

Status of the import error information file	Output destination	
	syslog file	Import error information file
N/A		
Normal (output enabled)	If true	Y
Error (output disabled)	If true	--

Y: Output.

If true: Output if `true` is specified in the `syslogout` operand in the import system definition.

--: Not output.

(8) Activity trace file handling

You can use the `hdstrcredit` command to view and edit activity trace files (import trace files). For details about how to use the `hdstrcredit` command, see the `hdstrcredit` command in Chapter 7. *Command Syntax*.

(9) Unimported information file handling

The following explains how to handle an unimported information file.

(a) Changing the maximum size

To change the maximum size of an unimported information file:

1. Terminate the source system.
2. Terminate the target Datareplicator.
3. Use a text editor to modify the corresponding operand in the import environment definition.
4. Execute initial start or partial initial start on the target Datareplicator.
5. Start the source system.

(b) Saving the unimported information file

When an unimported information file being used for data accumulation becomes full, file swapping occurs. Datareplicator re-creates the next unimported information file that is to be swapped in.

To save the contents of an unimported information file before the file is re-created:

1. Use an OS command to check the most recent update dates and times of the unimported information files.
2. Use an OS command to copy the unimported information file with the older update date and time into a new file under any name.

(c) Sending the unimported update information to the standard output

The unimported information file that is subject to accumulation of information is swapped when it become full. A new unimported information file is re-created when this swapping occurs. If you want to send the contents of the unimported information file to the standard output before the file is re-created, follow the procedure below:

1. Use the OS command to check the update dates and times of the unimported information files.
2. Send the unimported information file that has the older update date and time to the standard output.

(d) Example of unimported information file output

The following figure shows an example of the output from an unimported information file.

Figure 6-22: Example of output from an unimported information file

```
Fri Mar 29 09:29:38 2002 INSERT INTO "SHINYA"."SNDF1"("CKEY1", "CKEY2", "CKEY3",
"CCHAR1", "CNCHAR1", "CMCHAR1", "CINT1", "CSINT1", "CDEC1", "CLDEC1", "CFLT1", "CSFLT1",
"CDATE1", "CTIME1", "CIYD1", "CIHS1", "CBLOB1", "CADT1", "CMCOL1")
VALUES('CKEY1CKEY1', 1234567890, +19951003.,
'CHAR1 AAAA', 'NCHAR', 'MCHAR1 AAA', 305419896, 4660, 123456789012345,
12345678901234567890123456789, 1.317774742903815E-82, 1.317774742903815E-82,
'2001-12-24', '14:55:11', +12340102., +112233., **BLOB(4096)**
, SGMLTEXT(**BLOB(1024)**), ARRAY[10, 20])
```

Notes on the output format

- If a column's attribute is an abstract data type, Datareplicator uses a constructor function to import its data, and the update information value is output in the following format:

column-name=FUNC(update-information-value)

FUNC: Name of the constructor function

- If a column's attribute is repetition column and the update information indicates column-specified update processing, Datareplicator uses ARRAY to provide readable information in the following format (although actual update processing uses an iteration structure instead of ARRAY):
`column-name=ARRAY[update-information-value , update-information-value , . . . , update-information-value]`
- If a column's attribute is repetition column and the update information indicates element-specified update processing, Datareplicator outputs the information in the following format:
`column-name[element-number]=update-information-value`
- If BLOB concatenation operation data is updated, Datareplicator outputs the information in the following format:
`column-name=column-name | | *BLOB(data-length-in-bytes) *`
- If BINARY concatenation operation data is updated, Datareplicator outputs the information in the following format:
`column-name=column-name | | *BINARY(data-length-in-bytes) *`

(10) Command log file handling

The following explains how to handle a command log file.

The command log files contain a record of when Datareplicator commands were executed. The command log files are created automatically when Datareplicator starts. You can view a command log file at any time to check the execution history of commands. For details about the information that is output to the command log file, see *Overview of commands* in 7. *Command Syntax*.

The following figure shows an example of the contents of a command log file.

Figure 6-23: Example of command log file contents

```
Fri Aug 18 09:36:52 2006 pid=596 SYSTEM hdsstart : END : status=0, arg= -i.
Fri Aug 18 09:37:15 2006 pid=912 administrator hdsrfctl : END : status=0, arg= -d 01 -m spd.
Fri Aug 18 09:40:18 2006 pid=952 SYSTEM hdsstop : END : status=0, arg= -t force.
```

(11) Update information definition file handling

The following explains how to handle the update information definition file.

Modifying defined information

You can modify defined information before executing the `hdssamqin` command regardless of the target Datareplicator's status.

To modify defined information:

1. Use a text editor to modify the defined information.
2. Execute the `hdssamqin` command.

If the previous command has not terminated normally, do not modify the defined information. If you have modified defined information in such a case, execute the `hdssamqin` command with the `-c` option specified.

(12) SAM file handling

Transfer a SAM file created at a mainframe system that uses SAM files to the target Datareplicator or to a system that supports the update information input command (`hdssamqin`).

(13) Unextracted data storage file handling

The following explains how to handle an unextracted data storage file.

(a) Checking the unextracted data storage file

If data has been stored in an unextracted data storage file, Datareplicator issues a message to that effect when execution of the `hdssamqin` command finishes. When this message is output, check the contents of the file.

(b) Saving the unextracted data storage file

An unextracted data storage file is created each time the `hdssamqin` command is executed. To save the file contents, use an OS command to save the contents under a different filename before executing the next `hdssamqin` command.

(c) Example of unextracted data storage file output

The following figure shows an example of the output from an unextracted data storage file.

Figure 6-24: Example of output from an unextracted data storage file

```
DBM name = aaa, CDataset name = bbb, CSAM file offset = xx, Cinformation = Not defined
DBM name = aaa, CDataset name = bbb, CSAM file offset = xx, Cinformation = Invalid data length
DBM name = aaa, CDataset name = bbb, CSAM file offset = xx, Cinformation = Not defined
```

aaa: DBM name

bbb: Dataset name

xx: Location of update data in the SAM file (byte location from the beginning of the file).

Not defined: Update data in the dataset that has not been specified in an extraction statement.

Invalid data length: Update data that cannot be extracted as a variable-length record.

Note

Update data that cannot be extracted • indicates incomplete update data in a defined or redefined field.

6.7.3 Notes on handling the target Datareplicator

This section provides important information about handling the target Datareplicator.

(1) Notes on update processing

- Do not lock a table that is involved in import processing. If such a table is locked, import processing might result in the lock release wait status or deadlock.
- If a table at a target system is updated from multiple source systems, the order of the update processing cannot be predicted. If you need to control the order of update processing, you must do so by handling the processing appropriately. Suppose that data linkage is established from two source systems, A and B, to table T1 at a target system, and that a table row corresponding to source system A's mapping key value 100 is updated. Then a table row corresponding to source system B's mapping key value 100 is updated. In this case, the contents of the table row corresponding to key value 100 at the target system might not be the same as that of the corresponding row at source system B.
- Before you modify the extraction definition at the source system, you must ensure that all update information has been sent from the source system to the target Datareplicators. If you modify the extraction definition while some update information still remains untransmitted at the source Datareplicator, import processing might result in an error. In such a case, a target Datareplicator might not be able to recover the error, necessitating use of a program such as HiRDB Datareplicator to re-create tables.
- The table-based import method is always used when a UOC routine is used for import processing.

(2) Notes on initial start and partial initial start

- To initialize the import environment for a specified data linkage identifier only, you must execute partial initial start. To initialize the import environments for all data linkage identifiers, terminate import processing for all data linkage identifiers, and then execute initial start on the entire import environment.
- If you start the source system in the initialized status, start the target Datareplicator with the `hdsstart -i` or `hdsstart -i -D` command. If you start only the source system in the initialized status, the KFRB02003-E error results (detail code 5).

If you specified `nocheck` in the `extract_init` operand in the import environment definition, there is no need to start the target Datareplicator in the initial start or partial initial start mode.

(3) Notes on events

- Even if an event code sent from the source system is undefined in the import environment definition, there is no effect on the import processing; the target

Datareplicator still recognizes it as an event.

- While import processing is underway, the target Datareplicator ignores any transaction-based or table-based import restart event that is detected.

(4) Notes about tables for which the *WITHOUT ROLLBACK* option is specified

The following prerequisites and limitations apply to data linkage on tables for which the *WITHOUT ROLLBACK* option is specified.

Prerequisites:

The prerequisites listed below must all be satisfied. If any of the prerequisites is not satisfied, an inconsistency might occur between the source and target databases.

- The versions of the source and target Datareplicators are both 08-02 or later.
- The source and target databases are both HiRDB.
- `nodemst` is specified in the `sendcontrol` operand in the extraction system definition.
- If the source and target databases are on a parallel server, the table for which the *WITHOUT ROLLBACK* option is specified and all tables related to that table are stored on the same back-end server.

Limitations:

The following limitations apply to data linkage on tables for which the *WITHOUT ROLLBACK* option is specified:

- Data linkage is not supported for the following tables when the *WITHOUT ROLLBACK* option is specified:
 - Merge tables
 - Time-ordered information tables
 - Tables whose rows are partitioned among servers
 - Event control tables
- Only the `UPDATE SQL` statement can be used to perform data linkage. Because the `INSERT` statement cannot be used for data linkage, use a program such as HiRDB Dataextractor to create the target database.
- If the *WITHOUT ROLLBACK* option is specified for a table, its column specified for mapping keys cannot be updated.
- Import errors cannot be skipped. Specify SQL coding to skip such errors.
- The data linkage recovery facility is not supported. Recover data linkage errors individually.

(a) Notes about referencing the target database

For a table for which the `WITHOUT ROLLBACK` option is specified, one update operation is treated as one transaction. Therefore, if a table for which the `WITHOUT ROLLBACK` option is specified and a table for which the option is omitted are updated within the same transaction, the target Datareplicator sends and imports data as separate transactions. If the target database is referenced before all the transactions have been imported, an inconsistency might occur between the source and target databases. Therefore, make sure that the import processing has been completed[#] before you reference the target database.

Furthermore, if the target Datareplicator rolls back processing due to a failure, the contents of a table for which the `WITHOUT ROLLBACK` option is specified might not be refreshed temporarily while update information is being imported. Therefore, make sure that the import processing has been completed[#] before you reference the target database.

#

Make sure that `read position` and `write position` values that are displayed by executing the `hdsstate` are the same.

(b) Specification of the `sendintvl` operand in the transmission environment definition

For tables for which the `WITHOUT ROLLBACK` option is specified, one update operation is treated as one transaction. If 0 (send update information in units of transactions) is specified in the `sendintvl` operand in the transmission environment definition, transmission processing occurs each time a table for which the `WITHOUT ROLLBACK` option is specified is updated, resulting in a large overhead for transmission processing. Therefore, if you apply data linkage to tables for which the `WITHOUT ROLLBACK` option is specified, we recommend that you specify a nonzero value in the `sendintvl` operand in the transmission environment definition.

(5) Notes about tables for which the `COMPRESSED` option is specified

The following prerequisites apply to data linkage on tables for which the `COMPRESSED` option is specified.

Prerequisites:

The prerequisites listed below must all be satisfied. If any of the prerequisites is not satisfied, an inconsistency might occur between the source and target databases or import processing might result in an SQL execution error.

- The versions of the source and target Datareplicators are both 08-06 or later.
- The source and target databases are both HiRDB.
- The same endian is used for both the source and the target database.

6. Operation

- The compression specification is the same between the source and target tables.

6.8 Changing the configuration of HiRDB and Datareplicator

This section explains how to make HiRDB and Datareplicator configuration changes.

If you make changes to a system configuration as described in the table below, you must initialize the Datareplicator. If the Datareplicator is not initialized, the update information and configuration information retained in the Datareplicator might no longer match the actual information, and import processing might not be performed as you intended.

The following table describes the configuration changes that require initialization of a Datareplicator.

Table 6-14: Configuration changes that require initialization of a Datareplicator

Item		Description	Subsection
Changing the configuration of the source system	Changing the configuration of the source HiRDB servers	Changing the information specified in the <code>pdstart</code> and <code>pdunit</code> operands in the source HiRDB's system common definition	6.8.2
	Changing the source HiRDB's system log	<ul style="list-style-type: none"> Changing the number of system log files Initializing system log files Executing HiRDB's <code>pdlogchg -R</code> command to forcibly change the system log status 	
	Changing the definition of tables subject to linkage	<ul style="list-style-type: none"> Adding tables subject to linkage (<code>CREATE TABLE</code>) Deleting tables subject to linkage (<code>DROP TABLE</code>) Changing the tables subject to linkage (<code>ALTER TABLE</code>) 	
	Changing the source Datareplicator's definitions ^{#1}	<ul style="list-style-type: none"> Changing an operand whose definition does not take effect during normal start^{#2} Changing the contents of the duplexing definition file 	
Changing the configuration of the target system	Changing the definition of tables subject to linkage	<ul style="list-style-type: none"> Adding tables subject to linkage (<code>CREATE TABLE</code>) Deleting tables subject to linkage (<code>DROP TABLE</code>) Changing the tables subject to linkage (<code>ALTER TABLE</code>) 	6.8.3

Item		Description	Subsection
	Changing the source Datareplicator's definitions	<ul style="list-style-type: none"> Changing an operand whose definition does not take effect during normal start^{#2} Changing the contents of the duplexing definition file 	
	Changing the configuration of the target HiRDB servers ^{#3}	Changing the information specified in the <code>pdstart</code> and <code>pdunit</code> operands in the target HiRDB's system common definition	

Note:

For details about changing the configuration when the import transaction synchronization facility is used, see *6.8.4 Changing the configuration when the import transaction synchronization facility is used*.

#1

If you use the `hdemodq` command to add or delete extraction information queue files, there is no need to initialize the source or target Datareplicator.

#2

For details about operands, see the *Modifying defined information* section for each definition in Chapter 5. *Definitions*.

#3

Initialization is required if the key range-based partitioning method or the hash partitioning method is defined in the import group definition in the import definition. Otherwise, initialization is not required.

If you need to perform initialization or re-create the target database before all data linkage is completed, see *6.5.6(1) Handling procedure for re-creating the target database*.

6.8.1 Changing the definition of tables subject to linkage

The table below lists and describes the changes to tables that require initialization of a Datareplicator and modification of definitions. Note that changes to tables that are not subject to data linkage do not require Datareplicator initialization or modification of definitions.

Table 6-15: Changes to tables that require Datareplicator initialization and modification of definitions

Change to table	Whether Datareplicator initialization is required	Modification of extraction definition	Modification of import definition
Adding a table [#]	Y	Add the <code>extract</code> statement.	Add the <code>load</code> statement.
Deleting a table	Y	Delete the <code>extract</code> statement.	Delete the <code>load</code> statement.
Deleting a table and then re-creating the table (changing the table ID)	Y	--	--
Adding a non-mapping-key column to the target of linkage	Y	<p>When an asterisk (*) is specified in the definition of the columns that are to be extracted by using the <code>extract</code> statement: There is no need to modify the definition.</p> <p>When column names are specified in the definition of the columns that are to be extracted by using the <code>extract</code> statement: Add the column name.</p>	<p>When an asterisk (*) is specified in the definition of the fields that are to be imported by using the <code>load</code> statement or when the <code>load</code> statement is omitted: There is no need to modify the definition.</p> <p>When field names are specified in the definition of the columns that are to be extracted by using the <code>load</code> statement: Add the field name.</p>

6. Operation

Change to table	Whether Datareplicator initialization is required	Modification of extraction definition	Modification of import definition
Adding a mapping-key column to the target of linkage	Y	<p>When an asterisk (*) is specified in the definition of the columns that are to be extracted by using the <code>extract</code> statement: Add the column to the <code>key</code> or <code>ukey</code> clause. If the column is to be a transmission condition, add it to the <code>where</code> clause of the <code>send</code> statement.</p> <p>When column names are specified in the definition of the columns that are to be extracted by using the <code>extract</code> statement: Add the column name to the definition of the columns to be extracted and also add it to the <code>key</code> or <code>ukey</code> clause. If the column is to be a transmission condition, add it to the <code>where</code> clause of the <code>send</code> statement.</p>	<p>When an asterisk (*) is specified in the definition of the fields that are to be imported by using the <code>load</code> statement or when the <code>load</code> statement is omitted: There is no need to modify the definition.</p> <p>When field names are specified in the definition of the columns that are to be extracted by using the <code>load</code> statement: Add the field name.</p>
Removing a non-mapping key-column from the target of linkage	Y	<p>When an asterisk (*) is specified in the definition of the columns that are to be extracted by using the <code>extract</code> statement: There is no need to modify the definition.</p> <p>When column names are specified in the definition of the columns that are to be extracted by using the <code>extract</code> statement: Delete the column name.</p>	<p>When an asterisk (*) is specified in the definition of the fields that are to be imported by using the <code>load</code> statement or when the <code>load</code> statement is omitted: There is no need to modify the definition.</p> <p>When field names are specified in the definition of the columns that are to be extracted by using the <code>load</code> statement: Delete the field name.</p>

Change to table	Whether Datareplicator initialization is required	Modification of extraction definition	Modification of import definition
Removing a mapping key-column from the target of linkage	Y	<p>When an asterisk (*) is specified in the definition of the columns that are to be extracted by using the <code>extract</code> statement: Delete the column name from the <code>key</code> or <code>ukey</code> clause. If the column is used as a transmission condition, also delete it from the <code>where</code> clause of the <code>send</code> statement.</p> <p>When column names are specified in the definition of the columns that are to be extracted by using the <code>extract</code> statement: Delete the column name from the definition of the columns subject to extraction and also delete it from the <code>key</code> or <code>ukey</code> clause. If the column is used as a transmission condition, also delete it from the <code>where</code> clause of the <code>send</code> statement.</p>	<p>When an asterisk (*) is specified in the definition of the fields that are to be imported by using the <code>load</code> statement or when the <code>load</code> statement is omitted: There is no need to modify the definition.</p> <p>When field names are specified in the definition of the columns that are to be extracted by using the <code>load</code> statement: Delete the field name.</p>
Adding an RDAREA to a back-end server subject to linkage	N	--	--
Adding an RDAREA to a back-end server that was not subject to linkage	Y	<p>There is no need to modify the extraction definition. Modify the extraction environment definition as described below, and then initialize the source Datareplicator.</p> <ul style="list-style-type: none"> Specify in the <code>dsid</code> operand in the extraction environment definition a data linkage identifier (that is unique among all back-end servers) for the back-end server that will be subject to linkage. If <code>true</code> is specified in the <code>extsuppress</code> operand for the back-end server that will be subject to linkage, specify <code>false</code>. 	<p>Add the <code>load</code> statement to the import definition for the data linkage identifier that receives update information. Also, specify the following definitions in addition to the import definition:</p> <ul style="list-style-type: none"> Define in the <code>dsid</code> operand in the import system definition the data linkage identifier that will be receiving update information. Create the import environment definition for the data linkage identifier that will be receiving update information.

Change to table	Whether Datareplicator initialization is required	Modification of extraction definition	Modification of import definition
Deleting a column that is not subject to linkage	Y	--	--
Renaming a table	Y	Change the table identifier in the <code>extract</code> statement.	Change the table identifier in the <code>load</code> statement.
Renaming a column	Y	When an asterisk (*) is specified in the definition of the columns that are to be extracted by using the <code>extract</code> statement: There is no need to modify the definition. When column names are specified in the definition of the columns that are to be extracted by using the <code>extract</code> statement: Rename the column.	--
Changing a column's data type (column attribute)	Y	If the column is used as a transmission condition, change the condition in the <code>where</code> clause of the <code>send</code> statement.	--
Changing a partition storage condition	N	--	--

Legend:

Y: Datareplicator must be initialized.

N: There is no need to initialize Datareplicator.

--: There is no need to modify definitions.

#

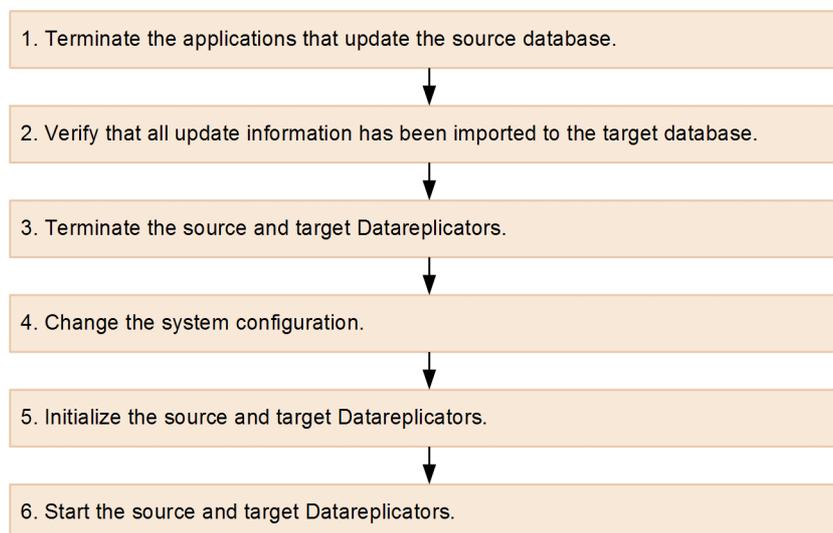
If you are adding tables that will be subject to extraction, check and, if necessary, revise the `extinforum` operand value in the extraction system definition. If the `extinforum` operand value is smaller than the number of `extract` statements, an error occurs during execution of the `hdeprep` command.

6.8.2 Changing the configuration of the source system

This subsection explains the initialization procedure to be used when the configuration of the source system is changed. If you are changing the configurations of both source and target systems, also use the procedure described here to initialize the Datareplicator.

The following figure shows the initialization procedure.

Figure 6-25: Procedure for changing the configuration of the source system



Note:

You must start the source and target databases to perform the steps beginning with step 5.

The procedure for changing the configuration of the source system shown in the figure is explained below. The numbers correspond to the numbers in the figure.

1. Terminate the applications that update the source database.
2. Verify that all update information has been imported to the target database.

No.	Task	Execution command	Check item
1	Verify that all processes are running at the source Datareplicator. If there is any inactive process, restart it.	hdestate	Verify that <code>Status</code> in the displayed results is not any of the following: ^{#1} <ul style="list-style-type: none"> • <code>init</code> • <code>hold</code> • <code>not active</code> • <code>not active (error)</code>
2	Verify that all processes are running at the target Datareplicator. If there is any inactive process, restart it.	hdsstate	Verify that <code>PID</code> in the displayed results is not <code>-1</code> . ^{#2}
3	Output the contents of all system logs to the system log file.	pdlogsync -d sys ^{#3}	Verify that the <code>KFPS02183-I</code> message has been output to the <code>syslog</code> file (event log in Windows).
4	Verify that all update information has been extracted.	pdls -d rpl -j	Check the information in the displayed results. ^{#4}
5	Verify that all update information has been transmitted.	hdestate	Verify that <code>current used ratio</code> in the displayed results is 0%. Alternatively, verify that <code>Queue write position</code> and <code>Queue read position</code> in the displayed results indicate the same position.
6	Verify that all update information has been imported.	hdsstate	Verify that <code>current used ratio</code> in the displayed results is 0%. Alternatively, verify that <code>Queue write position</code> and <code>Queue read position</code> in the displayed results indicate the same position.

#1

For a back-end server that is not subject to extraction, `not active` is displayed in `Status`.

#2

If `sendmst` is specified in the `sendcontrol` operand in the extraction system definition, `-1` might be displayed in `PID:hdstcpmst` in `COMMUNICATION INFORMATION`.

#3

For a HiRDB/Parallel Server, specify the server name in the `-s` option. If there are multiple back-end servers, execute the command for each and every back-end server that contains a table subject to extraction.

#4

As shown in the example below, verify that (1) is larger than (2) in the displayed results.

```

SYSTEMID      : HRD1(183346)
Data replication : Y
UNITID       : unt1(183346)
Data replication : Y
SERVER NAME  : sds01
Extract Database : Y
Extract Status : C
System Log Extract Point :
Run ID  Group  Gen No.  BLock No.
4740e7a9 log24  18      2dd8      ... (1)
System Log Sync Info :
Run ID  Group  Gen No.  BLock No.
4740e7a9 log24  18      2dba      ... (2)

```

3. Terminate the source and target Datareplicators.

No.	Task	Execution command	Check item
1	Terminate the source Datareplicator.	hdestop	--
2	Verify that the source Datareplicator has been terminated.	hdestate	Verify that the KFRB04411-E message has been output to the standard error output.
3	Terminate the target Datareplicator.	hdsstop	--
4	Verify that the target Datareplicator has been terminated.	hdsstate	Verify that the KFRB04302-E message has been output to the standard error output.

Legend:

--: Not applicable

4. Change the system configuration.

5. Initialize the source and target Datareplicators.

No.	Task	Execution command	Check item
1	Stop data linkage with HiRDB.	pdrplstop -f	Verify that the KFPS05141-I message has been output to the standard output.

6. Operation

No.	Task	Execution command	Check item
2	Initialize the source Datareplicator.	<code>hdestart -i</code> (Enter <code>Y</code> in the response message)	Verify that the <code>KFRB00504-I</code> message has been output to <code>msterrfile1</code> or <code>msterrfile2</code> .
3	Create the extraction definition preprocessing file.	<code>hdeprep -f</code> <i>extraction-definition-file#</i>	Verify that the <code>KFRB04500-I</code> message has been output to the standard output.
4	Start data linkage with HiRDB.	<code>pdrplstart</code>	Verify that the <code>KFPS05140-I</code> message has been output to the standard output.
5	Initialize the target Datareplicator.	<code>hdsstart -i -q#</code>	Verify that the <code>KFRB04216-I</code> message has been output to the standard output (Event Viewer in Windows).

#

If necessary, specify options.

6. Start the source and target Datareplicators.

No.	Task	Execution command	Check item
1	Start the target Datareplicator.	<code>hdsstart#</code>	Verify that the <code>KFRB00100-I</code> message has been output to <code>errfile1</code> or <code>errfile2</code> .
2	Start the source Datareplicator.	<code>hdestart#</code>	Verify that the <code>KFRB00502-I</code> message has been output to <code>msterrfile1</code> or <code>msterrfile2</code> .

#

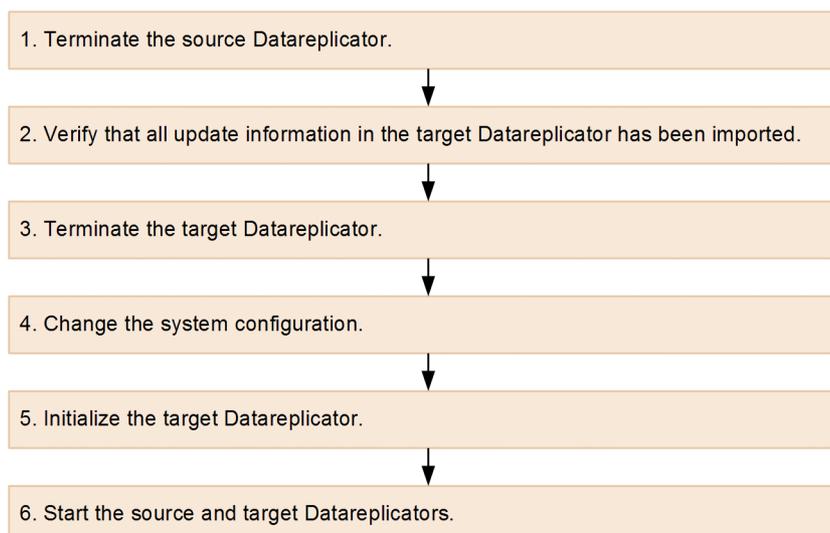
If necessary, specify options.

6.8.3 Changing the configuration of the target system

This subsection explains the initialization procedure to be used when the configuration of the target system is changed. If you are changing the configurations of both the source and target systems, see *6.8.2 Changing the configuration of the source system* for details about the initialization procedures.

The following figure shows the initialization procedure.

Figure 6-26: Procedure for changing the configuration of the target system



Note:

You must start the source and target databases to perform the steps beginning with step 5.

The procedure for changing the configuration of the target system shown in the figure is explained below. The numbers correspond to the numbers in the figure.

1. Terminate the source Datareplicator.

No.	Task	Execution command	Check item
1	Terminate the source Datareplicator.	<code>hdestop</code>	--
2	Verify that the source Datareplicator has been terminated.	<code>hdestate</code>	Verify that the <code>KFRB04411-E</code> message has been output to the standard error output.

Legend:

--: Not applicable

2. Verify that all update information in the target Datareplicator has been imported.

6. Operation

No.	Task	Execution command	Check item
1	Verify that all processes are running at the target Datareplicator. If there is any inactive process, restart it.	hdsstate	Verify that PID in the displayed results is not -1.#
2	Verify that all update information has been imported.	hdsstate	Verify that current used ratio in the displayed results is 0%. Alternatively, verify that Queue write position and Queue read position in the displayed results indicate the same position.

#

If sendmst is specified in the sendcontrol operand in the extraction system definition, -1 might be displayed in PID:hdstcpmst in COMMUNICATION INFORMATION.

3. Terminate the target Datareplicator.

No.	Task	Execution command	Check item
1	Terminate the target Datareplicator.	hdsstop	--
2	Verify that the target Datareplicator has been terminated.	hdsstate	Verify that the KFRB04302-E message has been output to the standard error output.

Legend:

--: Not applicable

4. Change the system configuration.

5. Initialize the target Datareplicator.

No.	Task	Execution command	Check item
1	Initialize the target Datareplicator.	hdsstart -i -q#	Verify that the KFRB04216-I message has been output to the standard output (Event Viewer in Windows).

#

If necessary, specify options.

6. Start the source and target Datareplicators.

No.	Task	Execution command	Check item
1	Start the target Datareplicator.	hdsstart [#]	Verify that the KFRB00100-I message has been output to errfile1 or errfile2.
2	Start the source Datareplicator.	hdestart [#]	Verify that the KFRB00502-I message has been output to msterrfile1 or msterrfile2.

#

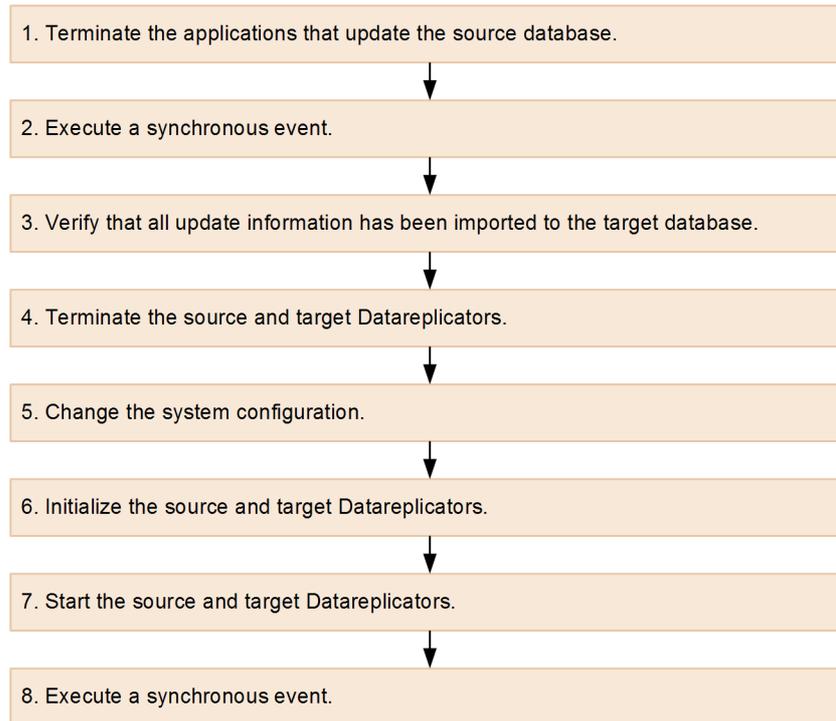
If necessary, specify the options.

6.8.4 Changing the configuration when the import transaction synchronization facility is used

This subsection explains the initialization procedure to be used when the import transaction synchronization facility is used.

The following figure shows the initialization procedure.

Figure 6-27: Procedure for changing the configuration when the import transaction synchronization facility is used



Note:

You must start the source and target databases to perform the steps beginning with step 6.

The procedure for changing the configuration when the import transaction synchronization facility is used that is shown in the figure is explained below. The numbers correspond to the numbers in the figure.

1. Terminate the applications that update the source database.
2. Execute a synchronous event.

No.	Task	Execution command	Check item
1	Execute a synchronous event to synchronize, and then terminate the import processing.	<code>hdevent -n event-code[#]</code>	Verify that no message has been output to the standard error output.

#

If necessary, specify options.

3. Verify that all update information has been imported to the target database.

No.	Task	Execution command	Check item
1	Verify that all processes are running at the source Datareplicator. If there is any inactive process, restart it.	hdestate	Verify that <code>Status</code> in the displayed results is not any of the following: ^{#1} <ul style="list-style-type: none"> • <code>init</code> • <code>hold</code> • <code>not active</code> • <code>not active (error)</code>
2	Verify that all processes are running at the target Datareplicator. If there is any inactive process, restart it.	hdsstate	Verify that <code>PID</code> in the displayed results is not <code>-1</code> . ^{#2}
3	Output the contents of all system logs to the system log file.	pdlogsync -d sys ^{#3}	Verify that the <code>KFPS02183-I</code> message has been output to the syslog file (event log in Windows).
4	Verify that all update information has been extracted.	pdls -d rpl -j	Check the information in the displayed results. ^{#4}
5	Verify that all update information has been transmitted.	hdestate	Verify that <code>current used ratio</code> in the displayed results is 0%. Alternatively, verify that <code>Queue write position</code> and <code>Queue read position</code> in the displayed results indicate the same position.
6	Verify that all update information has been imported.	hdsstate	Verify that <code>current used ratio</code> in the displayed results is 0%. Alternatively, verify that <code>Queue write position</code> and <code>Queue read position</code> in the displayed results indicate the same position.

#1

For a back-end server that is not subject to extraction, `not active` is displayed in `Status`.

#2

If `sendmst` is specified in the `sendcontrol` operand in the extraction system definition, `-1` might be displayed in `PID:hdstcpmst` in `COMMUNICATION INFORMATION`.

#3

For a HiRDB/Parallel Server, specify the server name in the `-s` option. If there are multiple back-end servers, execute the command for each and every back-end server that contains a table subject to extraction.

#4

As shown in the example below, verify that (1) is larger than (2) in the displayed results.

```

SYSTEMID      : HRD1(183346)
Data replication : Y
UNITID       : unt1(183346)
Data replication : Y
SERVER NAME   : sds01
Extract Database : Y
Extract Status : C
System Log Extract Point :
Run ID   Group   Gen No.  BLock No.
4740e7a9 log24   18       2dd8      ... (1)
System Log Sync Info :
Run ID   Group   Gen No.  BLock No.
4740e7a9 log24   18       2cba      ... (2)

```

4. Terminate the source and target Datareplicators.

No.	Task	Execution command	Check item
1	Terminate the source Datareplicator.	<code>hdestop</code>	--
2	Verify that the source Datareplicator has been terminated.	<code>hdestate</code>	Verify that the <code>KFRB04411-E</code> message has been output to the standard error output.
3	Terminate the target Datareplicator.	<code>hdsstop</code>	--
4	Verify that the target Datareplicator has been terminated.	<code>hdsstate</code>	Verify that the <code>KFRB04302-E</code> message has been output to the standard error output.

Legend:

--: Not applicable

5. Change the system configuration.
6. Initialize the source and target Datareplicators.

No.	Task	Execution command	Check item
1	Stop data linkage with HiRDB.	<code>pdrplstop -f</code>	Verify that the <code>KFPS05141-I</code> message has been output to the standard output.
2	Initialize the source Datareplicator.	<code>hdestart -i</code> (Enter <code>y</code> in the response message)	Verify that the <code>KFRB00504-I</code> message has been output to <code>msterrfile1</code> or <code>msterrfile2</code> .
3	Create the extraction definition preprocessing file.	<code>hdeprep -f</code> <i>extraction-definition-file#</i>	Verify that the <code>KFRB04500-I</code> message has been output to the standard output.
4	Start data linkage with HiRDB.	<code>pdrplstart</code>	Verify that the <code>KFPS05140-I</code> message has been output to the standard output.
5	Initialize the target Datareplicator.	<code>hdsstart -i -q#</code>	Verify that the <code>KFRB04216-I</code> message has been output to the standard output (in Windows, Event Viewer).

#

If necessary, specify options.

7. Start the source and target Datareplicators.

No.	Task	Execution command	Check item
1	Start the target Datareplicator.	<code>hdsstart#</code>	Verify that the <code>KFRB00100-I</code> message has been output to <code>errfile1</code> or <code>errfile2</code> .
2	Start the source Datareplicator.	<code>hdestart#</code>	Verify that the <code>KFRB00502-I</code> message has been output to <code>msterrfile1</code> or <code>msterrfile2</code> .

#

If necessary, specify options.

8. Execute a synchronous event.

No.	Task	Execution command	Check item
1	Execute a synchronous event to activate all processes in the target Datareplicator's synchronous import group.	<code>hdeevent -n</code> <i>event-code#</i>	<ul style="list-style-type: none"> Verify that no message has been output to the standard error output. Verify that the <code>KFRB03009-I</code> message has been output to the target system's error information file and that <code>reason</code> is <code>SYNC EVENT</code>.

6. Operation

#

If necessary, specify options.

6.9 Using the system switchover facility

Datareplicator can achieve *system switchover* by using HiRDB's system switchover facility. This facility swaps Datareplicator systems when HiRDB systems are swapped. You cannot swap only the Datareplicator systems as the response to a Datareplicator error. For details about handling of the system switchover facility in HiRDB, see the *HiRDB Version 9 System Operation Guide*.

The cluster software programs shown in the following table support the system switchover facility.

Table 6-16: Supported cluster software programs

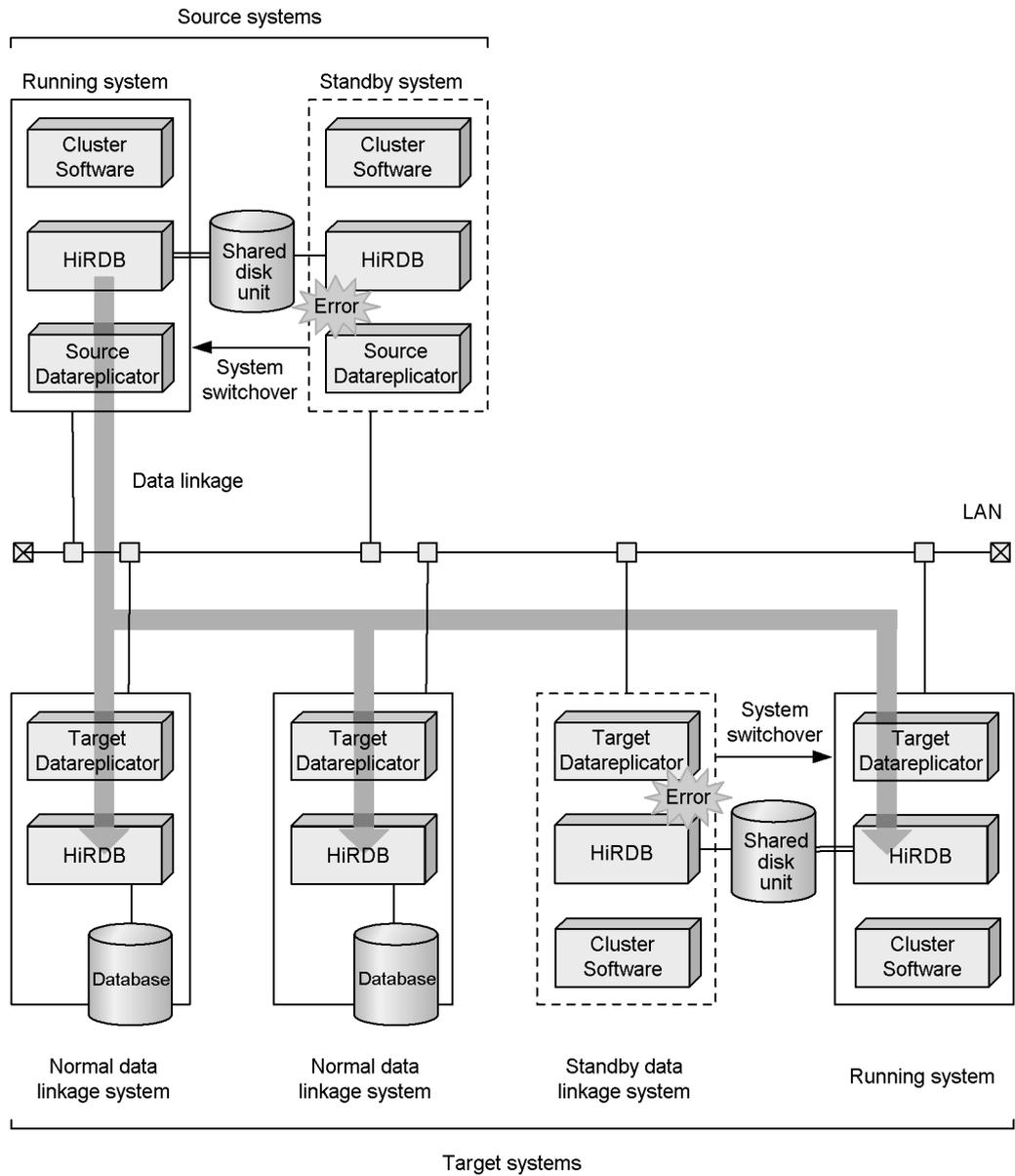
Cluster software	Applicable OS				
	HP-UX	Solaris	AIX	Linux	Windows
HA monitor	Y	--	Y	Y	--
MC/ServiceGuard	Y	--	--	--	--
HACMP	--	--	Y	--	--
Microsoft Cluster Server	--	--	--	--	Y

Y: Supported

--: Not supported

The following figure shows the data linkage system modes when the system switchover facility is used.

Figure 6-28: Data linkage system modes when the system switchover facility is used



6.9.1 System switchover facility modes

This section explains the system switchover facility modes.

(1) Types of system switchover

If you use the system switchover facility, make sure that with the HiRDB servers for the source Datareplicator you use *grouped-system switchover* (system switchover mode in which multiple products are grouped so that the entire group is subject to system switchover). For the target Datareplicator, use grouped-system switchover as needed.

This subsection describes the system switchover types supported by HiRDB. For details about each type of system switchover, see the related cluster software manual.

Automatic system switchover

When an error occurs in the running system, system switchover occurs automatically.

Planned system switchover

System switchover is implemented intentionally by the user by executing a cluster software command at the running system.

Grouped-system switchover can be implemented in either of the above two modes.

Other functions for reducing the HiRDB system switchover time include high-speed system switchover, user server hot standby, and transaction queuing.

(2) System configurations

The system switchover facility supports the following system configurations:

1-to-1 switchover organization

There is a one-to-one correspondence between a running system and a standby system.

2-to-1 switchover organization

There is a two-to-one correspondence between running systems and a standby system.

Mutual system switchover configuration

The same server machine contains both the running system and the standby system.

The mutual system switchover configuration requires a definition in order to use predefined error information files.

The following table lists the subsections that provide the details of different system switchover configurations.

Table 6-17: Subsections providing the details of different system switchover configurations

Cluster software used	System switchover configuration	Subsection
HA monitor	1-to-1 switchover configuration 2-to-1 switchover configuration	6.9.2 Preparations for using the system switchover facility (for HA monitor) 6.9.4 Handling procedure when the system switchover facility is used (for HA monitor)
	Mutual system switchover configuration	6.9.2 Preparations for using the system switchover facility (for HA monitor) 6.9.4 Handling procedure when the system switchover facility is used (for HA monitor) 6.9.5 Notes on using the system switchover facility
	Standby-less system switchover (effects distributed) facility	6.9.4 Handling procedure when the system switchover facility is used (for HA monitor) 6.9.6 Using the standby-less system switchover (effects distributed) facility
Microsoft Cluster Server	1-to-1 switchover configuration	6.9.3 Preparations for using the system switchover facility (for Microsoft Cluster Server)

6.9.2 Preparations for using the system switchover facility (for HA monitor)

This subsection describes the preparations that are required in order to use the HA monitor's system switchover facility. The items described here are applicable to all types of system switchover that are described in 6.9.1 (1) *Types of system switchover*.

(1) Preparations related to the HA monitor

You must use the HA monitor's `server` definition statement to set up the Datareplicator's operating environment. When linked with HiRDB, Datareplicator can execute system switchover. The following shows a specification example of the `server` definition statement:

Example of definition when the HA monitor's resource server facility is used

```

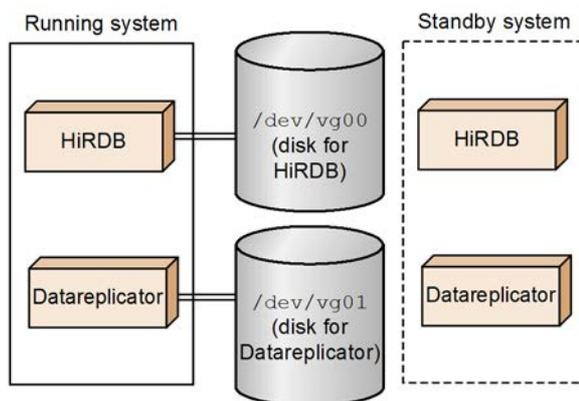
/**** Resource ****/
resource alias          REPres          ,
      group             groupA          ,
      initial           online          ,
      disk              /dev/vg00:/dev/vg01 ,
      lan_updown       use              ;

```

```

/**** Replicator ****/
server name /HAMon/etc/replica.up ,
      alias repl ,
      acttype monitor ,
      initial online ,
      termcommand /HAMon/etc/replica.down ,
      lan_updown nouse ,
      parent REPres ,
      group groupA ;

```

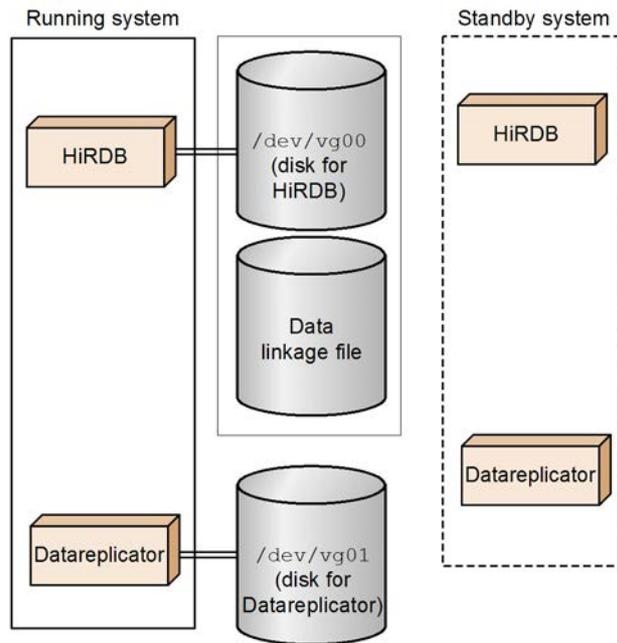


Example of definition when the HA monitor's resource server facility is not used

```

/**** Replicator ****/
server name /HAMon/etc/replica.up ,
      alias repl ,
      acttype monitor ,
      initial online ,
      termcommand /HAMon/etc/replica.down ,
      disk /dev/vg01 ,
      lan_updown use ,
      group groupA ;

```



To achieve source Datareplicator system switchover, the data linkage file is shared with the HiRDB. Therefore, you must adjust the volume layout in the data linkage file as well as the timing of starting HiRDB and Datareplicator.

If your HA monitor version supports the resource server facility, using the HA monitor's resource server facility for the source Datareplicator's system switchover eliminates the need to adjust the start timing. We recommend that you use the resource server facility in order to simplify the configuration of the system switchover environment.

The following describes each operand of the `server` definition statement that is related to Datareplicator.

(a) Resource server definition

The following describes each operand that is related to the resource server definition.

`alias`

Specify the identification name for the command to be used with the HA monitor or for the messages to be output.

`group`

Specify the same server group name as for the HiRDB and Datareplicator so that grouped system switchover is performed with HiRDB and Datareplicator.

`initial`

Specify the status when the server starts. For the primary system, specify `online`; for the secondary system, specify `standby`.

`disk`

Specify the path name of the volume group of the shared disk unit that contains the files requiring switchover for HiRDB and Datareplicator.

For details about the file to be created on the shared disk unit, see Table 6-18 *Preparation of files for using the system switchover facility*.

`lan_updown`

Specify `use`. This is because the IP address must be changed after system switchover so that communication can be established between the extraction master process and extraction node master process at the source Datareplicator.

Additionally, you must create the LAN status setup files for achieving the above IP address change (`server-ID.up` and `server-ID.down` files).

(b) Datareplicator-related definition

The following describes each operand that is related to the Datareplicator definition.

`name`

Specify the absolute path name of the shell specifying the command for starting Datareplicator. The following shows an example shell specification:

```
sleep 5# 1.
su - hircbdba -c /opt/hircbds/bin/hdestart 2.
                2> /tmp/start_log
```

`#`

If you use the resource server facility at the source or target Datareplicator, `sleep` is not needed.

1. HiRDB's `server` definition statement is used to swap disks for the data linkage file that is shared by HiRDB and Datareplicator. Taking into account the time lag before the data linkage file can be referenced after swapping HiRDB systems, set a wait time of some 5-10 seconds before Datareplicator is started.
2. The shell is executed from the HA monitor by the superuser, but the Datareplicator start command must be executed by the Datareplicator administrator. Therefore, the Datareplicator administrator (`hircbdba` in the above example) must start Datareplicator using the `su` command.

The start command for the source Datareplicator is different from that for the

target Datareplicator:

Source Datareplicator: `/opt/hirdbds/bin/hdestart`

Target Datareplicator: `/opt/hirdbds/bin/hdsstart`

Error messages during execution of the start command are output to the standard error output. If the start command is executed within the shell, set up in such a manner that the standard error output is routed to a work file (`/tmp/start_log` in the above example).

`alias`

Specify the identification name for the command to be used with the HA monitor or for the messages to be output.

`acttype`

Specify monitor.

`initial`

Specify the status when the server starts. For the primary system, specify `online`; for the secondary system, specify `standby`.

`termcommand`

Specify the absolute path name of the shell specifying the command for terminating Datareplicator when the system is to be terminated by the HA monitor's `monend` command alone or when planned system swapping is to be executed by the `monswap` command alone.

The following shows an example shell:

```
su - hirbdba -c "/opt/hirdbds/bin/hdestop
-t sendterm -w" 2> /tmp/stop_log 1.
```

1. Execute the `hdestop` command with `-w` specified because it is important to ensure that Datareplicator will terminate completely. The shell is executed from the HA monitor by the superuser, but the Datareplicator termination command must be executed by the Datareplicator administrator. Therefore, the Datareplicator administrator (`hirbdba` in the above example) must terminate Datareplicator using the `su` command.

The termination command for the source Datareplicator is different from that for the target Datareplicator:

Source Datareplicator: `/opt/hirdbds/bin/hdestop -t sendterm -w#`

Target Datareplicator: `/opt/hirdbds/bin/hdsstop -t force -w#`

#: For the `-t sendterm` and `-t force` options, change the specification as appropriate to the purpose, such as by specifying `-t immediate` for planned system switchover.

Error messages during execution of the stop command are output to the standard error output. If the stop command is executed within the shell, set up in such a manner that the standard error output is routed to a work file (`/tmp/stop_log` in the above example).

disk

Specify the file to be created on the shared disk unit. For details about the file to be created on the shared disk unit, see Table 6-18 *Preparation of files for using the system switchover facility*.

If the resource server facility is not used, specify the path name of the volume group of the shared disk unit that contains the files requiring switchover. Because the disk containing the data linkage file is also accessed from the HiRDB system, specify it in the `disk` operand of the `server` definition statement for the HiRDB system.

If you use the resource server facility, there is no need to specify this information because it has been specified in the resource server definition.

lan_updown

The specification is the same for the source and target systems.

Source system

Specify `use`. This is because the IP address must be changed after system switchover so that communication can be established between the extraction master process and extraction node master process.

Additionally, you must create the LAN status setup files for achieving the above IP address change (`server-ID.up` and `server-ID.down` files). Specify in the LAN status setup files the host name subject to IP address switchover control that was specified in the `node_host` operand in the Datareplicator's extraction system definition.

Target system

Specify `use`. This is because the IP address must be changed after system switchover so that communication can be re-established from the source Datareplicator.

Additionally, you must create the LAN status setup files for achieving the above IP address change (`server-ID.up` and `server-ID.down` files).

parent

Specify the identification name of the resource server (value specified in the

alias operand).

group

Specify the same server group name as for the HiRDB system so that linked system switchover is achieved with the HiRDB system. There is no need to specify the switchover type because Datareplicator is in the monitor mode.

(2) Hardware-related preparations

To use the system switchover facility, you must allocate the Datareplicator files to a shared disk unit. For details about the Datareplicator's files that need to be allocated, see *Table 6-18 Preparation of files for using the system switchover facility*.

Create the files listed in *Table 6-18 Preparation of files for using the system switchover facility* on an external hard disk (character special files) that is shared by the primary and secondary systems, and then define the files so that they can be accessed from both Datareplicators using the same path. You can use a Datareplicator file system area as the shared disk unit.

(3) Network-related preparations

It is necessary for the following networks to be configured in such a manner that they inherit IP addresses:

- Between Source Datareplicator's extraction master process and extraction node master process
- Link between source and target Datareplicators

(4) Preparation of files

To use the system switchover facility, you must allocate some files to the shared disk unit and copy in advance some other files from the primary system to the secondary system without changing the directory names.

For a file that is allocated to the shared disk unit, you need to create a symbolic link so that it can be referenced from the HiRDB Datareplicators on both the primary and the secondary systems using the same path name. For details about how to create a symbolic link, see *4.6.2 Preparation of the files used with the source Datareplicator* and *4.7.2 Preparation of the files used with the target Datareplicator*.

The following table shows the file-related preparations that are required when the system switchover facility is used.

Table 6-18: Preparation of files for using the system switchover facility

Classification	Filename	Shared disk unit	Same directory name?
Source Datareplicator	Extraction system definition file	N/A	Yes
	Extraction environment definition file	N/A	Yes
	Transmission environment definition files	N/A	Yes
	Extraction definition file [#]	N/A	N/A
	Duplexing definition file	N/A	Yes
	Duplexing control file	N/A	Yes
	Extraction definition preprocessing file	D	N/A
	Extraction information queue files	D	N/A
	Extraction master status file	D	N/A
	Extraction server status file	D	N/A
	Extraction master error information files	N/A	N/A
	Extraction node master error information files	N/A	N/A
	Extraction master trace files	N/A	N/A
	Extraction node master trace files	N/A	N/A
	Data linkage file	H	N/A
	Command log file	N/A	N/A

Classification	Filename	Shared disk unit	Same directory name?
Target Datareplicator	Import system definition file	N/A	Yes
	Import environment definition files	N/A	Yes
	Import definition files	N/A	Yes
	Duplexing definition file	N/A	Yes
	Duplexing control file	N/A	Yes
	Import information queue files	D	N/A
	Import status files	D	N/A
	Import master status file	D	N/A
	Import error information files	N/A	N/A
	Import trace files	N/A	N/A
	Unimported information files	N/A	N/A
	Command log files	N/A	N/A
	SAM files	N/A	Yes
	Update information definition file	N/A	Yes
	Unextracted data storage file	N/A	Yes
	User-created mapping table for converting character codes	N/A	Yes
Recovery information file	N/A	N/A	

D: The file must be allocated to the shared disk unit. The file must be allocated to a volume group on the shared disk unit that can be swapped when system switchover occurs on the HiRDB Datareplicator. Because the switchover timing is different, the file must be allocated to a different volume group than that for the file shown as H. The allocated volume group must be set by the HA monitor's server support environment setup so that it can be swapped when system switchover occurs on the HiRDB Datareplicator.

H: The file must be allocated to a volume group on the shared disk unit that can be swapped when system switchover occurs on the HiRDB Datareplicator. Because the switchover timing is different, the file must be allocated to a different volume group than that for the file shown as D. However, if the resource server facility is used, this file can be allocated to the same volume group as that for the file shown as D. The

allocated volume group must be set by the HA monitor's server support environment setup so that it can be swapped when system switchover occurs on the HiRDB Datareplicator.

Yes: The file must be allocated to the shared disk unit, or the file must be copied in advance from the primary system to the secondary system under the same directory name. If the file contents are updated, the updated file must also be copied to the secondary machine in advance.

N/A: Not applicable.

#: The file must be copied from the primary system to the secondary system in advance.

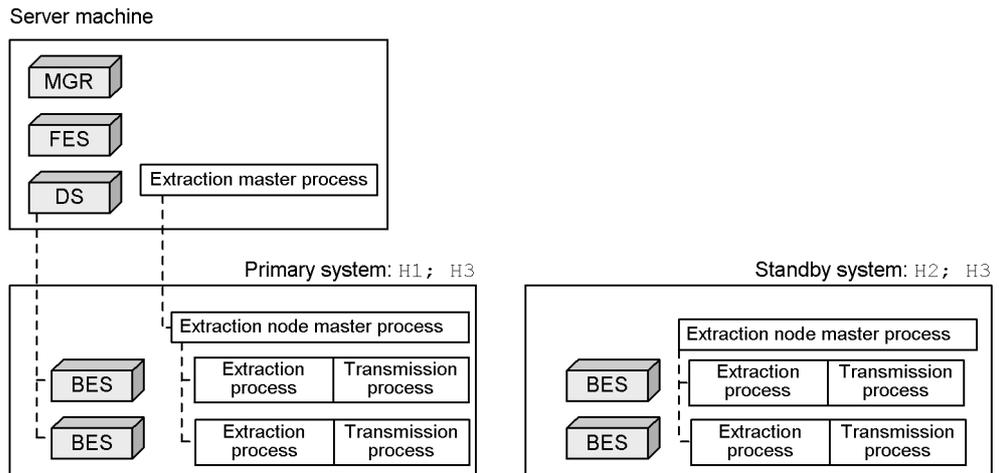
(5) How to define Datareplicator when the high-speed system switchover facility is used

For the source Datareplicator, you must allocate a host name that inherits the IP address for communication from the extraction master process to the extraction node master process. This host name must be different from the host name that is used by HiRDB.

You must specify this host name for the source Datareplicator that inherits IP addresses in the `node_host` operand of the extraction system definition in such a manner that it is associated with the host name specified in the `-x` operand of the `pdunit` command or the `pdstart` command in the source HiRDB's system common definition.

The following figure shows the relationship between the source system's server and main processes.

Figure 6-29: Relationship between the source system's server and main processes



H1, H2: Host names

H3: Host name corresponding to the IP address that is inherited between the systems

MGR: HiRDB system manager

DS: HiRDB dictionary server

FES: HiRDB front-end server

BES: HiRDB back-end server

▭ : Resident process

- - - - : Control

Define the correspondence between the host name specified in the `-x` operand of the `pdunit` command or the `pdstart` command in the source HiRDB's system common definition and the host name that starts the extraction node master process. The extraction master process connects to the host that starts the extraction node master process.

`pdsys` (system common definition)

```
pdunit -x H1 -c H2 -u unt1
```

`hdeenv` (extraction system definition)

```
nodedef (H1)
set node_host = H3
```

Connecting to the node master using H3

6.9.3 Preparations for using the system switchover facility (for Microsoft Cluster Server)

This subsection describes the preparations that are required in order to use the system switchover facility by Microsoft Cluster Server (MSCS). The items described here are applicable to all the types of system switchover that are described in 6.9.1 (1) *Types of*

system switchover.

For details about MSCS, see the MSCS documentation.

(1) **Limitations**

When MSCS's system switchover facility is used, only one unit is subject to extraction in both running and standby systems in the following cases:

- Multiple HiRDB units are running on the same server machine.
- In the mutual system switchover mode, multiple HiRDB units are able to run on the same server machine.

(2) **Preparations of Datareplicator**

In HDEPATH or HDSPATH for the primary and standby server machines, specify the pathname on the shared disk. The following limitations apply to the characters that can be used for path and directory names on the shared disk:

- Use either upper-case letters or lower-case letters consistently.
- Directory names can consist of only single-byte alphanumeric characters.
- Spaces, double-byte characters, and special symbols are not permitted.

Next, specify the system environment variables. The following table provides the details.

Table 6-19: Settings for the system environment variables (when MSCS's system switchover facility is used)

Classification	System environment variable	Setting in primary and standby systems
Source Datareplicator	HDEPATH	Path to the folder storing the extraction system definition file (hdeenv)
	PDDIR	PDDIR of the unit that contains MGR
	PDCONFPATH	PDCONFPATH of the unit that contains MGR
	PATH	Path to the source Datareplicator's command library [#]
Target Datareplicator	HDSPATH	Path to the folder storing the import system definition file (hdsenv)
	PDDIR	PDDIR of the target HiRDB
	PDNAMEPORT	Port number of the target HiRDB

Classification	System environment variable	Setting in primary and standby systems
	PDHOST	Host name of the target HiRDB
	PATH	Path to the target Datareplicator's command library [#]

#

Normally, this information is set automatically during installation of Datareplicator. If automatic setting results in an error due to the length of the PATH environment variable, you must specify this setting manually.

A client environment definition is required to execute the `hdeprep` and `hdeevent` commands at the source Datareplicator. For details about the client environment definition, see the *HiRDB Version 9 UAP Development Guide*.

(3) Service settings

The following table describes the Datareplicator service settings required to use the system switchover facility.

Table 6-20: Datareplicator service settings

Classification	Service name	Startup type	Registration into MSCS
Source Datareplicator	HiRDB Datareplicator (Source site)	Manual	Registered
	HiRDB Datareplicator (Source site NMT)	Automatic	Not registered
Target Datareplicator	HiRDB Datareplicator (Target Site)	Manual	Registered

(4) Hardware-related preparations

To use the system switchover facility, you must allocate the Datareplicator files to a shared disk unit.

Create all the files listed in Table 6-18 *Preparation of files for using the system switchover facility*, including the Datareplicator directory (specified in the `HDEPATH` and `HDSPATH` environment variables), on an external hard disk that is shared by the primary and secondary systems, and specify the settings so that both Datareplicators can reference the files using the same paths.

(5) MSCS-related preparations

The table below shows the resources that are to be registered into MSCS. For details

about how to perform the registration, see the MSCS documentation.

Table 6-21: Resources to be registered into MSCS

Classification	Definition item	Resource	Name	Dependency
Source Datareplicator	Shared disk used at the source Datareplicator ^{#1}	Shared disk	Any name	None
	Network name ^{#2}	Network name		
	IP address ^{#2}	IP address		
	Start service of the source Datareplicator	General-purpose service	HiRDB Datareplicator extract	Start service of HiRDB
Target Datareplicator	Shared disk used at the target Datareplicator ^{#3}	Shared disk	Any name	None
	Network name ^{#2}	Network name		
	IP address ^{#2}	IP address		
	Start service of the target Datareplicator	General-purpose service	HiRDB Datareplicator reflect	<ul style="list-style-type: none"> • Start service of HiRDB • Shared disk

#1

Add the shared disk as a dependency of *Start service of HiRDB*.

#2

Use the name set in HiRDB.

#3

Add the shared disk as a dependency of *Start service of the target Datareplicator*.

(6) Network-related preparations

Set the following network configuration elements to inherit IP addresses:

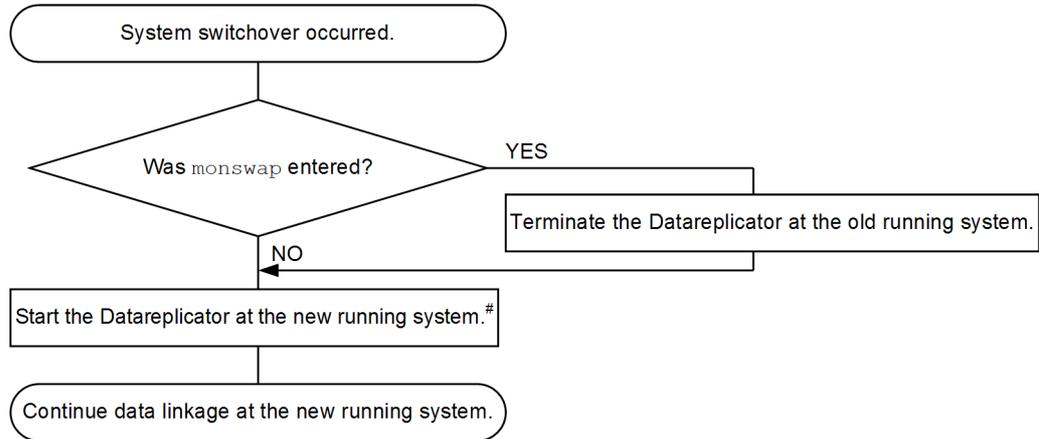
- Between the extraction master process and the extraction node master process of the source Datareplicator
- Between source and target Datareplicators

6.9.4 Handling procedure when the system switchover facility is used (for HA monitor)

The figure below shows the handling procedure when the system switchover facility is used with HA monitor.

For the handling procedure when the system switchover facility is used with MSCS, see the MSCS documentation.

Figure 6-30: Handling procedure when the system switchover facility is used with HA monitor



#

The source Datareplicator's commands must be issued to the extraction master process (`hdemaster`) that is running on HiRDB's system manager unit.

If the source HiRDB is a parallel server and system switchover occurs on a non-system manager unit, you must register a remote shell in the cluster software in such a manner that the Datareplicator's start and termination commands are issued to the system manager unit.

6.9.5 Notes on using the system switchover facility

This section provides important information about data linkage at a HiRDB system when the system switchover facility is used.

(1) When the source HiRDB is a parallel server and mutual system switchover configuration is used

When the source HiRDB is a parallel server and mutual system switchover configuration is used, system switchover might result in multiple active node master processes, all of which belong to the same source HiRDB on a single server machine. If this happens, each node master process's directory is duplicated on the same server

machine; as a result, filenames under the operation directory might be duplicated among the node master processes.

Figure 6-31 shows an example of a mutual system switchover configuration when the source HiRDB is a parallel server, and Table 6-22 shows the duplicate files in the operation directory based on server machine 3 in Figure 6-31.

Figure 6-31: Example of mutual system switchover configuration when the source HiRDB is a parallel server

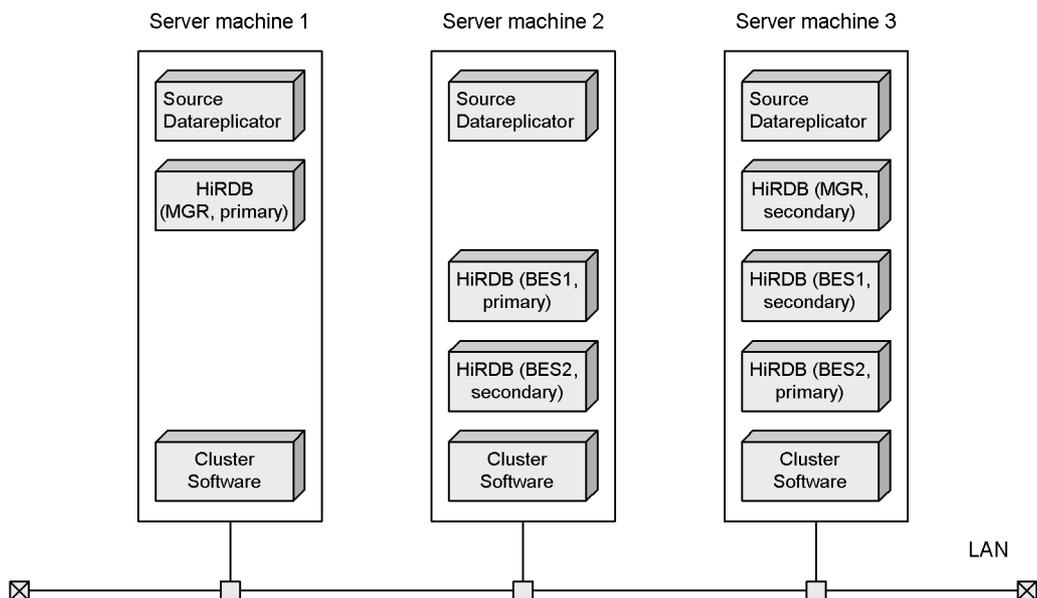


Table 6-22: Duplicate files in the operation directory based on server machine 3

No.	Active server			Files under Datareplicator directory
	MGR	BES1	BES2	
1	--	--	Y	<ul style="list-style-type: none"> • Queue files • Status files • Data linkage file • Error information files • Activity trace files
2	Y	--	Y	<ul style="list-style-type: none"> • Queue files • Status files • Data linkage file • Error information files • Activity trace files

No.	Active server			Files under Datareplicator directory
	MGR	BES1	BES2	
3	--	Y	Y	<ul style="list-style-type: none"> • Master status file • Master error information files • Queue files • Status files • Data linkage file • <u>Error information files</u> • <u>Activity trace files</u>
4	Y	Y	Y	<ul style="list-style-type: none"> • Master status file • Master error information files • Queue files • Status files • Data linkage file • <u>Error information files</u> • <u>Activity trace files</u>

Y: Active server

--: Inactive server

___ (underline): Files duplicated between node master processes

If multiple node master processes are active, the error information and activity trace files might be duplicated. When this type of system switchover configuration is used, specify `true` in the `errfile_unique` operand in the extraction system definition in order to prevent duplication of the error information files and the activity trace files. If you specify `false` in the `errfile_unique` operand and system switchover occurs with this organization, restart of the node master process will result in an error at the standby system.

(2) Monitoring the Datareplicator status

To monitor the Datareplicator's status, monitor the messages that are issued by the Datareplicator.

(3) When the disk needs to be activated manually

During system switchover, the cluster software controls activation and deactivation of the disk. However, if the disk is to be initialized after system switchover is stopped, the user must activate the disk manually. Before initializing the disk, make sure that you activate the disk. The disk cannot be initialized unless it is activated and made accessible prior to initialization.

(4) Notes about machine time settings

In a hot-standby configuration, synchronize the time settings in the running machine

and the standby machine. If their times are not synchronized when system switchover occurs, the extraction process cannot extract update information from the system log file.

6.9.6 Using the standby-less system switchover (effects distributed) facility

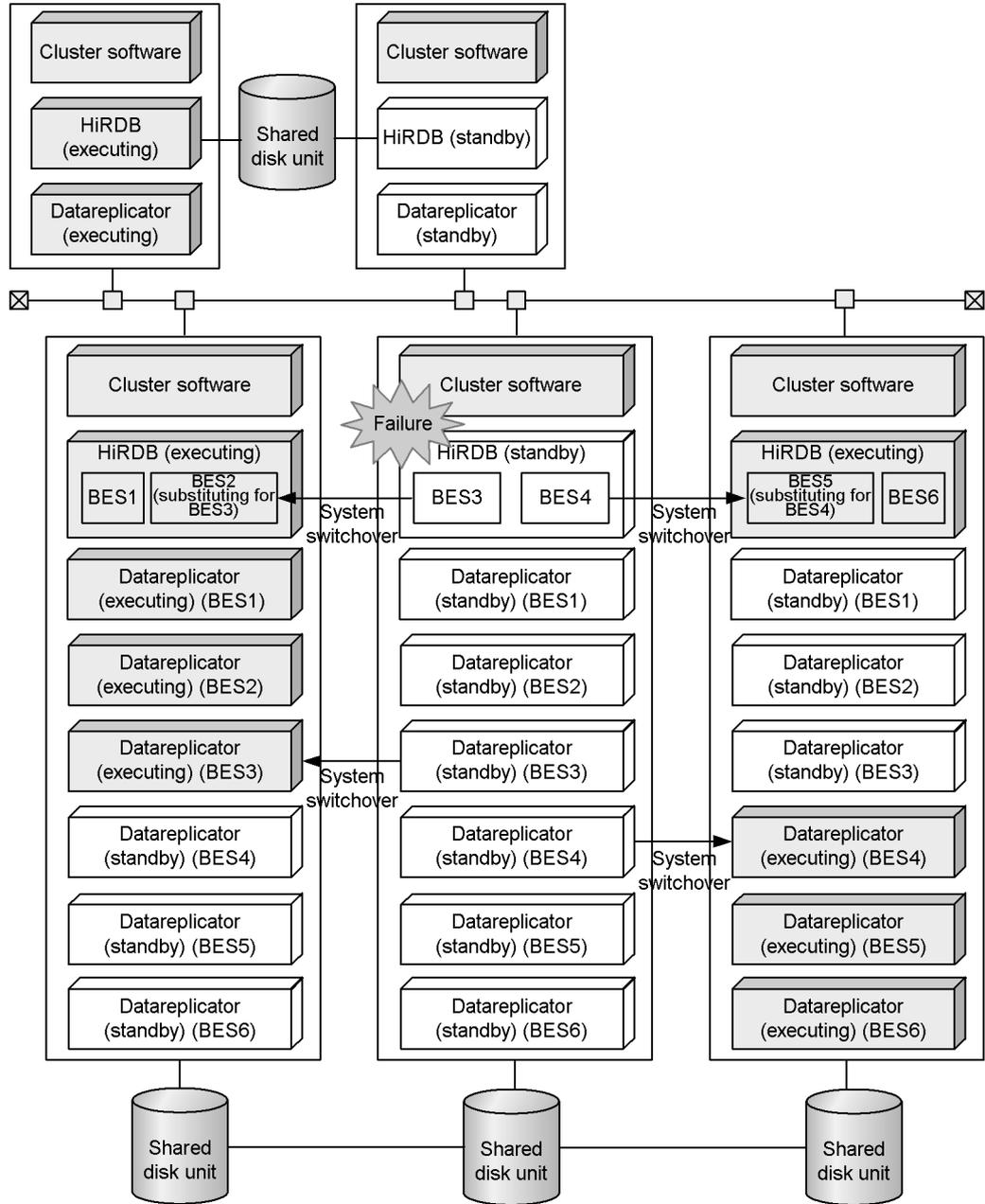
This subsection describes use of the standby-less system switchover (effects distributed) facility.

(1) Data linkage system modes when the standby-less system switchover (effects distributed) facility is used

If the source HiRDB employs a standby-less system switchover (effects distributed) configuration, the system is switched over for each back-end server. Therefore, settings for achieving system switchover for each back-end server are also needed for the source Datareplicator.

The following figure shows the data linkage system modes when the standby-less system switchover (effects distributed) facility is used.

Figure 6-32: Data linkage system modes when the standby-less system switchover (effects distributed) facility is used



(2) Preparations for using the standby-less system switchover (effects distributed) facility

This subsection describes the preparations that are required in order to use the standby-less system switchover (effects distributed) facility (this subsection covers only those items that are different from when the normal system switchover facility is used). For details about the preparations required in order to use the normal system switchover facility, see 6.9.2 *Preparations for using the system switchover facility (for HA monitor)*.

(a) Preparations related to the HA monitor

In the same manner as when normal system switchover is performed, use the HA monitor's `server` definition statement to define the Datareplicator operating environment. Because standby-less system switchover (effects distributed) takes place for each back-end server at the source HiRDB, you must also specify a Datareplicator definition for each of the HiRDB's back-end servers so that system switchover is linked with HiRDB.

For details, see 6.9.2 *Preparations for using the system switchover facility (for HA monitor)*.

(b) Allocating the source Datareplicator's host names

When you use the standby-less system switchover (effects distributed) facility, allocate host names as follows:

Host where the extraction master process is run:

Allocate a host name that inherits IP addresses. This must be a new host name for Datareplicator that is allocated separately from the host name used by the source HiRDB.

Host where the extraction node master process is run:

There is no need to allocate a new host name for Datareplicator because the host name without IP address inheritance that is used by HiRDB is also used by Datareplicator.

(c) Creating Datareplicator startup and termination shells

Terminate Datareplicator on the primary HiRDB and create a shell for starting Datareplicator on the standby HiRDB. The following shows an example of such a shell.

- Example of startup shell
- For the unit that contains the system manager

```
sleep 5#
su -user-name -c "/opt/hirbds/bin/hdestart_n" 2> /tmp/start_log
```

- For a unit that does not contain the system manager

```
sleep 5#
su -user-name -c "/opt/hirdbds/bin/hdestart_n -b BES-name -x host-name -n service-name "
2> /tmp/start_log
```

Note:

Error messages are output to the standard error output during execution of the start command. If you execute the start command within the shell, specify settings so that the standard error output is routed to a work file (in the above example, /tmp/start_log).

#

Adjust `sleep` in such a manner that Datareplicator is started after the volume containing the data linkage file is activated (after HiRDB has started).

However, if you are simplifying the source Datareplicator operation, `sleep` is not needed. For details about simplifying the operation of the source Datareplicator, see 6.9.6(3) *Simplifying the operation of the source Datareplicator (applicable only when HA monitor is used)*.

- Example of termination shell
- For the unit that contains the system manager

```
su -user-name -c "/opt/hirdbds/bin/hdestop_n -w" 2> /tmp/stop_log
```

- For a unit that does not contains the system manager

```
su -user-name -c "/opt/hirdbds/bin/hdestop_n -b BES-name -t sendterm -c continue -w" 2>
/tmp/stop_log
```

Note:

Error messages are output to the standard error output during execution of the termination command. If you execute the termination command within the shell, specify settings so that the standard error output is routed to a work file (in the above example, /tmp/stop_log).

(d) Extraction system definition

To use the standby-less system switchover (effects distributed) facility, you must set the following operands in the extraction system definition:

```
nodecontrol=server
```

When the standby-less system switchover (effects distributed) facility is used, HiRDB performs system switchover for each back-end server. Therefore, the source Datareplicator must also be set up to perform system switchover for each

back-end server in the same manner as in HiRDB.

`node_connection_accept=true`

Specifies that a connection request from the extraction node master process is to be accepted, and that connection is to be re-established between the extraction master process and the extraction node master process during system switchover.

`connection_accept_hostname=host-name-accepting-connection-request-from-extraction-node-master-process`

Specifies the host name with IP address inheritance that has been allocated to the server machine where the extraction master process is run in order to accept a connection request from the extraction node master process.

`connection_accept_service=service-name-accepting-connection-request-from-extraction-node-master-process`

Specifies the service name for accepting a connection request from the extraction node master process.

`connection_accept_waittime=extraction-node-master-process-connection-request-wait-time`

Specifies the amount of time to wait for the extraction master process to accept a reconnection request from the extraction node master process in the event all extraction node master processes are disconnected.

`connection_retry_time=extraction-node-master-process-reconnection-processing-time`

Specifies the amount of time to allot for the extraction node master process to reconnect to the extraction master process in the event that the extraction master process is disconnected.

(3) Simplifying the operation of the source Datareplicator (applicable only when HA monitor is used)

If both of the following conditions are satisfied, you can use the user command interface provided by the HA monitor:

- The cluster software in use is the HA monitor.
- The source HiRDB uses the standby-less system switchover (effects distributed) facility.

The user command interface provided by the HA monitor can simplify the preparation and handling of system switchover, because the source Datareplicator's system switchover control is included in HiRDB's system switchover control.

The following describes the preparations and handling of this simplification.

(a) Preparations

To prepare for system switchover:

1. Create a user command.

Create a user command for the startup and termination of Datareplicator that occur during HiRDB system switchover.

For an example of a user command, see *Example of a user command* below.

2. Register the user command for HA monitor environment settings.

Register the created user command into the `usrcommand` operand of the HA monitor environment settings so that the user command will execute during HiRDB system switchover.

3. Allocate shared resources.

- Allocating a host name with IP address inheritance

The resources to be allocated depend on the active process and the host name used by HiRDB. The following table shows whether shared resources are to be allocated:

Host name used by HiRDB	Process running at the host	
	Extraction master process	Extraction node master process [#]
IP addresses are inherited	N	N
IP addresses are not inherited	Y	N

Legend:

Y: A new host name that inherits IP addresses must be allocated for Datareplicator.

N: There is no need to allocate a new host name that inherits IP addresses.

#

For the network between the source and target Datareplicators, a host name that inherits IP addresses must be allocated for the target Datareplicator in the same manner as when the operation is not simplified.

- Allocating shared volumes

Not only the data linkage files but also all Datareplicator-related files must be allocated to the shared volumes that can be switched over by HiRDB's system switchover control.

4. Create the Datareplicator startup and termination shells.

To simplify operation, you must execute the Datareplicator startup and termination shells from the user command. For details about creating shells, see 6.9.6(2)(c) *Creating Datareplicator startup and termination shells*.

5. Create the cluster software definition.

Because Datareplicator's switchover control is achieved by the user command during HiRDB system switchover, there is no need to create an HA monitor server definition for Datareplicator's system switchover.

6. Specify the Datareplicator definition.

Same as when the operation is not simplified.

Example of a user command

- Adjust the value for each pattern in the `case` command according to the `server` definition for the HA monitor in use. Also adjust the number of patterns according to the number of HiRDB servers in use.
- Change the names of the Datareplicator startup and termination shells to the actual shell names that will be used (commands ending with `_sh` are the Datareplicator startup and termination shells).

The following shows an example of a user command:

```
#!/bin/sh
HA_SERVER_ID=$2
SERVER_KIND=$4
SERVER_STATUS=$5
START_PARAM=$6

#####
###
# Convert from HA monitor's server name to HiRDB's server
name #
#####
###
case "$HA_SERVER_ID" in
"REPmng" )
  HiRDB_SERVER_ID="MST"
  ;;
"REPb11" )
  HiRDB_SERVER_ID="bes11"
  ;;
"REPb12" )
  HiRDB_SERVER_ID="bes12"
  ;;
"REPb21" )
  HiRDB_SERVER_ID="bes21"
```

```

;;
"REPB22" )
  HiRDB_SERVER_ID="bes22"
  ;;
"REPB31" )
  HiRDB_SERVER_ID="bes31"
  ;;
"REPB32" )
  HiRDB_SERVER_ID="bes32"
  ;;
*)
  exit 0
  ;;
esac

#####
###
# Issue HiRDB Datareplicator command #
#####
###
if [ $SERVER_KIND = "online" ]
then
  case "$SERVER_STATUS" in
    "-s" )
      #####
      # CASE : online : Normal Start #
      #####
      if [ $START_PARAM = "end" ]
      then
        if [ $HiRDB_SERVER_ID = "MST" ]
        then
          /HiRDBDS/SH/StartMST_sh
        else
          /HiRDBDS/SH/StartNMT_sh $HiRDB_SERVER_ID "initial"
        fi
      fi
      ;;
    "-e" )
      #####
      # CASE : online : Normal End #
      #####
      if [ $START_PARAM = "start" ]
      then
        if [ $HiRDB_SERVER_ID = "MST" ]
        then
          /HiRDBDS/SH/StopMST_sh
        else

```

```

        /HiRDBDS/SH/StopNMT_sh $HiRDB_SERVER_ID "initial"
    fi
fi
;;
"-p")
#####
# CASE :   online : Planned End                               #
#####
if [ $START_PARAM = "start" ]
then
    if [ $HiRDB_SERVER_ID = "MST" ]
    then
        /HiRDBDS/SH/StopMST_sh
    else
        /HiRDBDS/SH/StopNMT_sh $HiRDB_SERVER_ID "initial"
    fi
fi
;;
"-a")
#####
# CASE :   online : Switch by Server down                     #
#####
if [ $START_PARAM = "start" ]
then
    if [ $HiRDB_SERVER_ID = "MST" ]
    then
        /HiRDBDS/SH/StopMST_sh
    else
        /HiRDBDS/SH/StopNMT_sh $HiRDB_SERVER_ID "continue" "-t
sendterm"
    fi
fi
;;
"-w")
#####
# CASE :   online : Switch by Planned                         #
#####
if [ $START_PARAM = "start" ]
then
    if [ $HiRDB_SERVER_ID = "MST" ]
    then
        /HiRDBDS/SH/StopMST_sh
    else
        /HiRDBDS/SH/StopNMT_sh $HiRDB_SERVER_ID "continue"
    fi
fi
;;
esac

```

```

else
case "$SERVER_STATUS" in
"-a")
#####
# CASE : standby : Switch by Server down #
#####
if [ $START_PARAM = "end" ]
then
if [ $HiRDB_SERVER_ID = "MST" ]
then
/HiRDBDS/SH/StartMST_sh
else
/HiRDBDS/SH/StartNMT_sh $HiRDB_SERVER_ID "continue"
fi
fi
;;
"-w")
#####
# CASE : standby : Switch by Planned #
#####
if [ $START_PARAM = "end" ]
then
if [ $HiRDB_SERVER_ID = "MST" ]
then
/HiRDBDS/SH/StartMST_sh
else
/HiRDBDS/SH/StartNMT_sh $HiRDB_SERVER_ID "continue"
fi
fi
;;
esac
fi

```

(b) Operation

1. Initializing the source Datareplicator

The following describes how to initialize the source Datareplicator:

1. Terminate normally the source HiRDB subject to extraction processing.
2. Terminate the HA monitor.
3. In the HA monitor environment settings, delete the specification of the Datareplicator control command from the `usrcommand` operand.
4. Start the HA monitor.
5. Start the source HiRDB in the unlinked mode (`pd_rpl_init_start=N`).
6. Initialize the source Datareplicator environment (`hdestart -i` command).

7. Create the extraction definition preprocessing file (`hdeprep` command).
8. Terminate the source HiRDB normally.
9. Terminate the HA monitor.
10. In the HA monitor environment settings, register the Datareplicator control command in the `usrcommand` operand.
11. Start the HA monitor.
12. Start the source HiRDB in the linked mode (`pd_rpl_init_start=Y`). The source Datareplicator is started at the same time HiRDB is started.

2. Starting and terminating the source Datareplicator

No operation is needed, because the source Datareplicator's startup and termination are linked to startup and termination of the source HiRDB.

3. Starting the source Datareplicator while HiRDB is terminated

If the source HiRDB cannot be started for a reason such as a log file that has become filled with unextracted log information, you can use the following procedure to start the source Datareplicator only:

1. Manually activate the shared resources in the system manager unit that are required to start Datareplicator.
2. Start the extraction master process on the system manager unit (`hdestart_n` command).
3. Manually activate the shared resources that are required for the extraction node master process that is to be started.
4. At the back-end server unit, start the extraction node master process (`hdestart_n -b server-name`).

6.10 Using the file duplexing function

This section describes how to modify the file organization for file duplexing. It describes the three configuration modifications listed below. In this section, normal file handling is called file non-duplexing, as opposed to file duplexing.

- Changing the file organization from non-duplexing to duplexing
- Changing the files to be duplexed
- Changing the file organization from duplexing to non-duplexing

For details about the changes applicable when Datareplicator Extension is used, see the manual *HiRDB Datareplicator Extension Version 8*.

6.10.1 Changing the file organization from non-duplexing to duplexing

This subsection describes the procedure for changing the file organization from non-duplexing to duplexing. The **bold type** indicates a duplexing-related step.

(1) Procedure at the source Datareplicator

This subsection describes the procedure at the source Datareplicator.

1. Terminate the source HiRDB normally.
2. Use the `hdestate` command to make sure that the `read` and `write` locations are the same in the extraction information queue file.
3. Use the `hdestop` command to terminate Datareplicator.
4. **Create the duplexing definition file.**
5. **Prepare the physical file of the file to be duplexed.**[#]
6. **Delete any file that has the same name as the logical file of the file to be duplexed.**
7. Use the `hdestart -i` command to initialize Datareplicator.
8. Make a backup of the duplexing control file created under `$HDEPATH` at each node.
9. If a system switchover configuration is used, copy the duplexing control file under `$HDEPATH` at the target.
10. Start the source HiRDB.
11. Execute the `hdeprep` command.
12. Use the `hdestart` command to start Datareplicator.

#

Preparing the physical file means allocating a partition to use a character special file, and then creating its symbolic link.

(2) Procedure at the target Datareplicator

This subsection describes the procedure at the target Datareplicator.

The basic difference from the procedure for the source Datareplicator is that there is no need to terminate HiRDB.

1. Use the `hdsstate` command to make sure that the `read` and `write` locations are the same in the import information queue file.
2. Use the `hdsstop` command to terminate Datareplicator.
3. **Create the duplexing definition file.**
4. **Prepare the physical file of the file to be duplexed.**[#]
5. **Delete any file that has the same name as the logical file of the file to be duplexed.**
6. Use the `hdsstart -i` command to execute initial startup of Datareplicator.
7. Make a backup of the duplexing control file created under `$HDSPATH`.
8. If a system switchover configuration is used, copy the duplexing control file under `$HDEPATH` at the target.

#

Preparing the physical file means allocating a partition to use a character special file, and then creating its symbolic link.

When you perform initialization for the first time after creating the import information queue file and the import status file as character special files, use the `hdsstart -i -f` command to perform an initial start of the target Datareplicator. There is no need to specify the `-f` option thereafter.

6.10.2 Changing the files to be duplexed

This subsection describes the procedure for adding, changing, and deleting a file to be duplexed. The **bold type** indicates the duplexing-related step.

This procedure is basically the same as for changing files from non-duplexing to duplexing.

(1) Procedure at the source Datareplicator

This subsection describes the procedure at the source Datareplicator.

1. Terminate the source HiRDB normally.

2. Use the `hdestate` command to make sure that the `read` and `write` locations are the same in the extraction information queue file.
3. Use the `hdestop` command to terminate Datareplicator.
4. **Change the duplexing definition file.**
5. **Prepare the physical file of the file to be duplexed.#**
6. **Delete the physical file that will no longer be used.**
7. **Delete any file that has the same name as the logical file of the file to be duplexed.**
8. Use the `hdestart -i` command to initialize Datareplicator.
9. Make a backup of the duplexing control file created under `$HDEPATH` at each node.
10. If a system switchover configuration is used, copy the duplexing control file under `$HDEPATH` at the target.
11. Start the source HiRDB.
12. Execute the `hdeprep` command.
13. Use the `hdestart` command to start Datareplicator.

#

Preparing the physical file means allocating a partition to use a character special file, and then creating its symbolic link.

(2) Procedure at the target Datareplicator

This subsection describes the procedure at the target Datareplicator.

The basic difference from the procedure for the source Datareplicator is that there is no need to terminate HiRDB.

1. Use the `hdsstate` command to make sure that the `read` and `write` locations are the same in the import information queue file.
2. Use the `hdsstop` command to terminate Datareplicator.
3. **Change the duplexing definition file.**
4. **Prepare the physical file of the file to be duplexed.#**
5. **Delete the physical file that will no longer be used.**
6. **Delete any file that has the same name as the logical file of the file to be duplexed.**
7. Use the `hdsstart -i` command to execute initial startup of Datareplicator.

8. Make a backup of the duplexing control file created under `$HDSPATH`.
9. If a system switchover configuration is used, copy the duplexing control file under `$HDEPATH` at the target.

#

Preparing the physical file means allocating a partition to use a character special file, and then creating its symbolic link.

When you perform initialization for the first time after creating the import information queue file and the import status file as character special files, use the `hdsstart -i -f` command to perform an initial start of the target Datareplicator. There is no need to specify the `-f` option thereafter.

6.10.3 Changing the file organization from duplexing to non-duplexing

This subsection describes the procedure for changing the file organization from duplexing to non-duplexing. The **bold type** indicates a step for changing duplexing to non-duplexing.

(1) Procedure at the source Datareplicator

This subsection describes the procedure at the source Datareplicator.

1. Terminate the source HiRDB normally.
2. Use the `hdestate` command to make sure that the `read` and `write` locations are the same in the extraction information queue file.
3. Use the `hdestop` command to terminate Datareplicator.
4. **Delete the physical file that has been used for duplexing.**
5. **Delete the duplexing control file created under `$HDEPATH` at each node.**
6. **If a system switchover configuration is used, delete the duplexing control file under `$HDEPATH` at the target.**
7. **Delete the file duplexing specification from the extraction system definition.**
8. Use the `hdestart -i` command to initialize Datareplicator.
9. Start the source HiRDB.
10. Execute the `hdeprep` command.
11. Use the `hdestart` command to start Datareplicator.

(2) Procedure at the target Datareplicator

This subsection describes the procedure at the target Datareplicator.

Use the `hdsstate` command to make sure that the `read` and `write` locations are the

same in the import information queue file.

1. Use the `hdsstop` command to terminate Datareplicator.
2. **Delete the physical file that has been used for duplexing.**
3. **Delete the duplexing organization definition file created under \$HDSPATH.**
4. **Delete the file duplexing specification from the import system definition.**
5. **Delete the duplexing control file created under \$HDSPATH.**
6. **If a system switchover configuration is used, delete the duplexing control file under \$HDEPATH at the target.**
7. Use the `hdsstart -i` command to execute initial startup of Datareplicator.

6.11 Handling of large files

This section describes how to handle extraction information queue files and import information queue files whose maximum size is at least 2 GB (large files). In this section, the extraction information queue files and import information queue files are referred to generically as queue files.

In the Windows edition

To use large files in the Windows edition, specify 2097152 (2 GB) or a greater value in the `queuesize` operands in both the extraction environment definition and the import environment definition.

In the UNIX edition

To use large files in the UNIX edition, specify 2097152 (2 GB) or a greater value in the `queuesize` operands in both the extraction environment definition and the import environment definition, and specify appropriate OS settings as well as the maximum size of the Datareplicator file system area. For details, see *6.11.1 Preparations for handling large files (UNIX edition only)*.

Note:

If the size of an existing queue file has been changed, Datareplicator must be initialized. Therefore, before you change the file size, verify that replication has been completed.

6.11.1 Preparations for handling large files (UNIX edition only)

This subsection describes the preparations for handling large files in the UNIX edition

(1) *Setting the maximum file size supported by the system*

The maximum file size supported by the system depends on the OS. Therefore, you must set the system to support large files by using one of the methods described below:

- Set the supported maximum file size to a value that is greater than the maximum file size used by the Datareplicator processes.
- Do not set a maximum value (set it to be unlimited).

For details about the OS commands required for changing the maximum value and how to use them, see the OS and shell documentation.

Note that in AIX, the default file size is 1 GB. To check the maximum file size supported by each OS, use the `limit` or `ulimit` command. For details, see the applicable OS and shell documentation.

Note:

In the AIX edition, the maximum file size used by a process that is started from `inetd` is the same as for the administrator privilege (superuser). Because the extraction master process is started from `inetd`, use the procedure described in the table below to change the maximum value for the administrator privilege (superuser). For details about the communication settings, see *2.5.1 Setting up a source Datareplicator's communications environment*.

Step	Setting	Command
1	Add the service name and port number that are used for communication between extraction master processes and extraction node processes.	# vi# /etc/services
2	Set the communication environment for <code>inetd</code> .	# vi# /etc/inetd.conf
3	Set the maximum file size for the administrator privilege (superuser).	# vi# /etc/security/limits
4	Terminate <code>inetd</code> .	# stopsrc -s inetd
5	Start <code>inetd</code> .	# startsrc -s inetd

#

Any text editor can be used.

(2) Setting the maximum size of a Datareplicator file system area

To store a queue file with a size of 2 GB or greater in a Datareplicator file system area, set 2 GB or a greater value in the `hdsfmkfs -n` command. Note that the maximum permissible size depends on the OS; check the following table for the applicable value.

Table 6-23: Maximum size of Datareplicator file system area when large files are used

Datareplicator type	Condition		Maximum size of Datareplicator file system area (MB)
HP-UX edition	Large files not used	Regular file	2,047
		Character special file	
	Large files used	Regular file	131,071
		Character special file	
Solaris edition	Large files not used	Regular file	2,047
		Character special file	

Datareplicator type	Condition		Maximum size of Datareplicator file system area (MB)
	Large files used	Regular file	1,048,575
		Character special file	
AIX edition [#]	Large files not used	Regular file	2,047
		Character special file	
	Large files used	Regular file (JFS)	65,411
		Regular file (JFS2)	1,048,575
		Character special file	
Linux edition	Large files not used	Regular file	2,047
		Character special file	
	Large files used	Regular file	1,048,575
		Character special file	

#

When you use large files, the permitted maximum size of a Datareplicator file system area is different between JFS and JFS2. Therefore, use the `smit` command to specify the necessary settings so that large files can be used in the target file system.

(3) Notes about setting link files

To handle large files using link names, you can specify only one nesting link. If more than one link is nested, as shown in the figure below, the target file cannot be created as a large file.



Legend:

→ : Symbolic link

Linkname1: Link name 1

Linkname2: Link name 2

REG_LARGE: File to be created

In this example, use Linkname2, not Linkname1.

6.11.2 Estimating the command execution time when large files are used

When queue files are handled as large files, the execution times of the following commands are increased:

- `hdestart` (start the source Datareplicator)
- `hdemodq` (modify the organization of extraction information queue files)
- `hdsstart` (start the target Datareplicator)
- `hdefcopy` and `hdsfcopy` (copy the current file)

(1) Estimation formula for `hdestart` (start the source Datareplicator)

When the `-i init` option is specified, the command execution time becomes longer.

Initial creation time for extraction information queue file (microseconds) =
 $\sum unit(T_UNIT)$
 $T_UNIT = \sum svr(T_SVR)$
 $T_SVR = \sum que(T_QUE)$
 $T_QUE = DiskW \times \text{size of queue file (KB)} / 1024$

- $\sum unit$
Sum of the extraction information queue file creation time in units
- T_UNIT
Initial creation time for all extraction information queue files per unit
- $\sum svr$
Sum of the extraction information queue file creation times at back-end servers

- *T_SVR*
Initial creation time for all extraction information queue files per back-end server
- Σque
Sum of the extraction information queue file creation times
- *T_QUE*
Initial creation time for one extraction information queue file
- *DiskW*
Time required for writing 1 KB of data on the disk where the files are placed

Note:

During initialization, specify the maximum value of the initial creation time for all extraction information queue files per unit (T_UNIT) as the communication wait time in the extraction system definition (cmwaittime operand value).

If processing is not terminated within the communication wait time in the extraction system definition (cmwaittime operand value) and an error (KFRB00607-E) occurs, increase the communication wait time, and then re-execute the command.

(2) Estimation formula for hdemodq (modify the organization of extraction information queue files)

When the -a option is specified, the command execution time becomes longer.

Additional registration time for extraction information queue file (microseconds) = DiskW x size of queue file (KB)/1024

- *DiskW*
Time required for writing 1 KB of data on the disk where the files are placed

(3) Estimation formula for hdsstart (start the target Datareplicator)

When the -i init option is specified, the command execution time becomes longer.

*Initial creation time for import information queue file (microseconds) = $\Sigma que(T_QUE) + \Sigma sts(T_STS)$
 $T_QUE = DiskW \times size\ of\ queue\ file\ (KB)/1024$
 $T_STS = DiskW \times size\ of\ status\ file\ (KB)/1024$*

- Σque

Sum of the import information queue file creation times

- T_{QUE}

Initial creation time for one import information queue file

- Σsts

Sum of the status file creation times

- T_{STS}

Initial creation time for one status file

- $DiskW$

Time required for writing 1 KB of data on the disk where the files are placed

(4) Estimation formula for hdefcopy and hdsfcopy (copy the current file)

Current file copying time (microseconds)
 = $DiskR \times \text{size of source file (KB)}/1024$
 + $DiskW \times \text{size of target file (KB)}/1024$

- $DiskR$

Time required for reading 1 KB of data from the disk where the files are placed

- $DiskW$

Time required for writing 1 KB of data onto the disk where the files are placed

Note:

The size of a file to be copied is determined as follows:

File type	File to be copied	Size
Regular file	All	Size of source file during command execution
Character special file	Extraction information queue file	<code>queuesize</code> operand value specified in the extraction environment definition
	Import information queue file	<code>queuesize</code> operand value specified in the import environment definition
	Other	Size of source file during command execution

6.12 Tuning

This section explains the tuning of Datareplicator.

6.12.1 Whether tuning is needed

Determine whether tuning is needed based on the following criteria:

- Source Datareplicator's transmission performance
- Target Datareplicator's import performance

(1) *Source Datareplicator's transmission performance*

To collect information about and determine whether the transmission performance needs to be tuned:

1. Periodically execute the `hdestate` command to collect information about `Queue write position` and `Queue read position`.
2. Obtain the value by subtracting the offset position of `Queue read position` from the offset position of `Queue write position`.
3. If the value obtained in step 2 has increased in proportion to the elapsed time, evaluate the following tuning items:
 - Transmission interval
 - Size of update information editing buffer
 - Transmission master process's transmission interval
 - Length of update information editing buffer
 - Interval at which the transmission process reads the extraction information queue file
 - Size of extraction information queue I/O buffer

Note that if the import information queue file is full or a communication line error occurs, the value obtained in step 2 also increases in proportion to the elapsed time. If this is the case, eliminate the cause of the error, obtain the offset positions again, and then evaluate the change of offset and the elapsed time.

(2) *Target Datareplicator's import performance*

To collect information about and determine whether the import performance needs to be tuned:

1. Periodically execute the `hdsstate` command to collect information about `Queue write position` and `Queue read position`.

6. Operation

2. Obtain the value by subtracting the offset position of `Queue read position` from the offset position of `Queue write position`.
3. If the value obtained in step 2 has increased in proportion to the elapsed time, evaluate the following tuning items:
 - SQL performance
 - Import processing commit interval
 - Import processing commit interval when the transaction-based import method is used
 - Import processing commit interval when the table-based import method is used
 - Maximum number of update SQL statements in an import transaction
 - COMMIT issuance interval
 - Concurrent execution of import processing
 - Interval at which the import process reads the import information queue file

You can determine SQL performance by collecting HiRDB's SQL trace. To collect an SQL trace, you need to specify the `PDSQLTRACE` environment variable. In UNIX, specify the `PDSQLTRACE` environment variable in the target Datareplicator's environment variables. In Windows, specify this environment variable in the system environment variables or `hirdb.ini`. For details about `PDSQLTRACE`, see the manual *HiRDB Version 9 System Definition*.

6.12.2 How to tune

The table below explains how to tune a Datareplicator. Choose the appropriate method after you have determine what needs to be tuned.

Table 6-24: Tuning methods

Processing performance	Elements of tuning	Tuning method	Recommended level
Transmission performance	<ul style="list-style-type: none"> • Transmission interval • Size of update information editing buffer • Transmission master process's transmission interval • Length of update information editing buffer 	Adjust the following operand values: <ul style="list-style-type: none"> • When <code>nodemst</code> is specified in the <code>sendcontrol</code> operand <ul style="list-style-type: none"> • <code>sendintvl</code> operand • <code>sendintvl_scale</code> operand • <code>editbufsize</code> operand For details, see <i>5.4 Transmission environment definition</i>. • When <code>sendmst</code> is specified in the <code>sendcontrol</code> operand <ul style="list-style-type: none"> • <code>smt_sendintvl</code> operand • <code>smt_sendintvl_scale</code> operand • <code>smt_editbufsize</code> operand For details, see <i>5.2 Extraction system definition</i>. 	Medium
	<ul style="list-style-type: none"> • Interval at which the transmission process reads the extraction information queue file • Size of extraction information queue I/O buffer 	Adjust the following operand values: <ul style="list-style-type: none"> • <code>quiosize</code> operand • <code>queue_read_wait_interval</code> operand For details about the <code>quiosize</code> operand, see <i>5.3 Extraction environment definition</i> . For details about the <code>queue_read_wait_interval</code> operand, see <i>5.4 Transmission environment definition</i> .	Low
Import performance	SQL performance	Do the following at the target HiRDB: <ul style="list-style-type: none"> • When multiple indexes have been defined for a target table Delete any unneeded index. • When the time required for import processing has increased Re-organize the RDAREA storing the target table. 	High

Processing performance	Elements of tuning	Tuning method	Recommended level
	<ul style="list-style-type: none"> • Import processing commit interval • Import processing commit interval when the transaction-based import method is used • Import processing commit interval when the table-based import method is used • Maximum number of update SQL statements in an import transaction 	Adjust the following operand values: <ul style="list-style-type: none"> • <code>cmtintvl</code> operand • <code>trncmtintvl</code> operand • <code>tblcmtintvl</code> operand • <code>reflect_trn_max_sqlnum</code> operand For details, see <i>5.9 Import environment definition</i> .	High
	COMMIT issuance interval	Adjust the <code>commit_wait_time</code> operand. For details, see <i>5.9 Import environment definition</i> .	High
	Concurrent execution of import processing	Define an import group and apply the table-based import method. For details, see <i>5.10 Import definition</i> .	Medium
	Interval at which the import process reads the import information queue file	Adjust the <code>ref_wait_interval</code> operand value. For details, see <i>5.9 Import environment definition</i> .	Low

6.13 Notes about operation

6.13.1 Notes about changing the OS time

If the OS time is moved back on the machine on which HiRDB Datareplicator is running, correct operation cannot be performed. To move back the OS time, see the *HiRDB System Operation Guide*.

Chapter

7. Command Syntax

This chapter explains the Datareplicator's command syntax. The commands are explained in alphabetical order.

Overview of commands

- hdechgstatus (change the status of the source Datareplicator)
- hdeevent (issue an event at the source Datareplicator)
- hdefcopy (copy the current source file)
- hdefstate (display the status of duplexed source files)
- hdemodq (modify the organization of extraction information queue files)
- hdeprep (create an extraction definition preprocessing file)
- hdeshmclean (delete the source Datareplicator's shared resources)
- hdestart (start the source Datareplicator)
- hdestart_n (partially start the source Datareplicator)
- hdestate (collect source Datareplicator status information)
- hdestop (terminate the source Datareplicator)
- hdestop_n (partially terminate the source Datareplicator)
- hdsagtopt (manipulate Datareplicator agent settings)
- hdsagtstart (start the Datareplicator agent)
- hdsagtstatus (display the Datareplicator agent status)
- hdsagtstop (terminate the Datareplicator agent)
- hdscnvedt (edit a mapping table for converting character codes)
- hdschgstatus (change the status of the target Datareplicator)
- hdsfcopy (copy the current target file)
- hdsfmkfs (initialize a Datareplicator file system area)
- hdsfstate (display the status of duplexed target files)
- hdsfstatfs (display the status of a Datareplicator file system area)
- hdsfpathlist (specify a directory to be monitored)
- hdsrefinfm (check update information)
- hdsrftcl (control import processing)
- hdssamqin (extract update information from a SAM file)
- hdsshmclean (delete the target Datareplicator's shared resources)
- hdsstart (start the target Datareplicator)
- hdsstate (collect target Datareplicator status information)
- hdsstop (terminate the target Datareplicator)
- hdstrcredit (edit an activity trace file)
- pdlogchg (modify the status of log-related files)
- pdls (display the HiRDB system status)

pdrplstart (start HiRDB Datareplicator linkage)
pdrplstop (cancel HiRDB Datareplicator linkage)

Overview of commands

This section explains the syntax provided for the commands that you can use to execute data linkage applications with Datareplicator.

Before you can use Datareplicator commands, you must specify environment variables. For details about the environment variables that need to be specified, see 2.4 *Specifying environment variables (UNIX)* or 2.8 *Specifying environment variables (Windows)*.

Executing commands on Windows Vista or Windows Server 2008

If the UAC feature is used, the processing depends on whether you run the command prompt as an administrator. For details, see 2.10.1 *Executing commands*.

Table 7-1 *Commands used to establish data linkage with Datareplicator* lists the commands that are used to establish data linkage with Datareplicator. Table 7-2 *Information that is output to the command log* lists the information that is output to the command log.

Table 7-1: Commands used to establish data linkage with Datareplicator

Command type	Command	Function	When executable	
			During HiRDB operation	During Datareplicator operation
Commands for handling the source Datareplicator	hdestart ^{#1}	Starts the source Datareplicator. When the <code>-i</code> option is specified, initializes the source Datareplicator environment but does not start the source Datareplicator.	N/A	Inactive OK ^{#4}
	hdestop ^{#2}	Terminates the source Datareplicator.	N/A	Only active
	hdeevent ^{#2}	Issues an event.	Only active	N/A
	hdestate ^{#2}	Outputs the source Datareplicator's status to the standard output.	N/A	Only active
	hdeprep ^{#2}	Analyzes the extraction definition and creates an extraction definition preprocessing file.	Only active	Inactive OK ^{#5}

Command type	Command	Function	When executable	
			During HiRDB operation	During Datareplicator operation
	hdemodq ^{#3}	Changes the organization of the extraction information queue files.	N/A	Only inactive
	hdefcopy ^{#2}	In the event of a file duplexing error, copies the current file into the erroneous file.	N/A	Only inactive
	hdefstate ^{#2}	Displays the status of the physical files in use for file duplexing.	N/A	Inactive OK
	hdeshmclean	Deletes shared resources at the source Datareplicator.	N/A	Only inactive
	hdestart_n ^{#2}	Partially starts the source Datareplicator.	N/A	Only active
	hdestop_n ^{#2}	Partially terminates the source Datareplicator.	N/A	Only active
	hdechgstasus ^{#2}	Changes the status and information that is maintained beyond the termination or start of the source Datareplicator.	N/A	Only active
Commands for handling the target Datareplicator	hdsstart	Starts the target Datareplicator. When the -i option is specified, initializes the target Datareplicator environment but does not start the target Datareplicator.	N/A	Only inactive
	hdsstop ^{#5}	Terminates the target Datareplicator.	N/A	Only active
	hdsrfctl ^{#5}	Controls the import processing method, or restarts import processing only.	N/A	Only active
	hdsstate ^{#5}	Outputs the target Datareplicator's status to the standard output.	N/A	Only active

Command type	Command	Function	When executable	
			During HiRDB operation	During Datareplicator operation
	hdssamqin ^{#6, #7}	Outputs to the import information queue file update information extracted from a mainframe database that uses SAM files.	N/A	Only active
	hdsfcopy ^{#6}	In the event of a file duplexing error, copies the current file into the erroneous file.	N/A	Only inactive
	hdsfstate ^{#6}	Displays the status of the physical files in use for file duplexing.	N/A	Inactive OK
	hdsrefinfm ^{#6}	Extracts each item of update information from the import information status file and outputs it to the analysis results output file.	N/A	Inactive OK
	hdschgstatus ^{#6}	Changes the status and information that is maintained beyond the termination or start of the target Datareplicator.	N/A	Only active
	hdsshmclean	Deletes shared resources at the target Datareplicator.	N/A	Only inactive
HiRDB commands	pdrplstart	The source HiRDB starts HiRDB Datareplicator linkage.	Only active	N/A
	pdrplstop	The source HiRDB cancels HiRDB Datareplicator linkage.	Only active	N/A
	pdl (-d rpl specified)	Displays the HiRDB Datareplicator linkage status at the source HiRDB.	N/A	N/A
	pdlogchg (-R specified)	Changes the system log file from extracting status to extraction completed status.	N/A	N/A
Commands for JPI/Cm2 operations management ^{#8}	hdsagtstart	Starts the Datareplicator agent.	N/A	N/A

Command type	Command	Function	When executable	
			During HiRDB operation	During Datareplicator operation
	<code>hdsagtstop</code>	Terminates the Datareplicator agent.	N/A	N/A
	<code>hdsagtstatus</code>	Displays the Datareplicator agent's status.	N/A	N/A
	<code>hdsagtopt</code>	Changes Datareplicator agent settings. The information to be specified depends on the options that are specified.	N/A	N/A
	<code>hdspathlist</code> ^{#10}	Specifies the Datareplicator directory on a machine that is subject to monitoring.	N/A	N/A
Command for editing a mapping table for converting character codes	<code>hdscnvedt</code>	Edits a mapping table for converting character codes. To update a mapping table for converting character codes or to migrate a Gaiji character mapping file, the superuser must execute this command.	N/A	N/A
Command for editing activity trace files	<code>hdstrcedit</code> ^{#2, #6}	Edits an activity trace file.	N/A	N/A
Commands for managing Datareplicator file system areas ^{#9}	<code>hdsfmkfs</code> ^{#2, #6, #10}	Initializes a Datareplicator file system area.	N/A	Only inactive
	<code>hdsfstatfs</code> ^{#2, #6, #10}	Displays the status of a Datareplicator file system area.	N/A	N/A

Only active: Executable only while HiRDB or Datareplicator is active.

Only inactive: Executable only while HiRDB or Datareplicator is inactive.

Inactive OK: Can be executed while Datareplicator is inactive.

N/A: Not applicable.

#1: When you execute the `hdestart` command, you must use a user name that belongs to the same user group as the HiRDB administrator.

#2: To execute this command, use the user name that was used to initialize the source Datareplicator (`hdestart -i` command).

#3: When you execute this command, use the name of the user who executes the extraction node master process (specified in `inetd.conf`).

#4: You cannot execute the `hdestart` command with the `-i` option specified while the source Datareplicator is active.

#5: When the `hdeprep` command is executed, the execution results do not become effective at the source Datareplicator until the source Datareplicator is restarted (the command's execution results do not become effective during the session in which the command is executed).

#6: To execute this command, use the user name that was used to initialize the target Datareplicator (`hdsstart -i` command).

#7: The `hdssamqin` command is available only when the source database uses SAM files. The target Datareplicator does not support the `hdssamqin` command when XDM/DS is used for data linkage.

#8: These commands are applicable to a Datareplicator that supports JP1/Cm2 operations management. Some of these commands can be executed only by the superuser. These commands are common to both source and target Datareplicators. For details about how to use JP1/Cm2 for operations management, see *3.4 Using JP1/Cm2 for operations management*.

#9: These commands are applicable to a Datareplicator that uses Datareplicator file system areas. Only the superuser can execute these commands. These commands are common to both source and target Datareplicators. For details about file management when Datareplicator file system areas are used, see *3.5 Datareplicator file system areas*.

#10: This command is not supported by Windows Datareplicator.

Table 7-2: Information that is output to the command log

Command name	Information that is output		
	Command's arguments	Command's return value	Details in the event of an error
<code>hdechgstatus</code>	N	N	N
<code>hdeevent</code> <code>hdeeventO</code> <code>hdeeventS</code>	Y	Y	N
<code>hdefcopy</code>	Y	Y	N
<code>hdefstate</code>	N	N	N
<code>hdemodq</code>	Y	Y	N

Command name	Information that is output		
	Command's arguments	Command's return value	Details in the event of an error
hdeprep hdeprepO hdeprepS	Y	Y	N
hdeshmclean	Y	Y	N
hdestart hdestartO hdestartS	Y	Y	Y
hdestart_n	Y	Y	Y
hdestate	N	N	N
hdestop hdestopO hdestopS	Y	Y	Y
hdestop_n	Y	Y	Y
hdsagtopt	N	N	N
hdsagtstart	N	N	N
hdsagtstatus	N	N	N
hdsagtstop	N	N	N
hdscnvedt	N	N	N
hdschgstatus	N	N	N
hdsfcopy	Y	Y	N
hdsfmkfs	N	N	N
hdsfstate	N	N	N
hdsfstatfs	N	N	N
hdspathlist	N	N	N
hdsrefinfm	N	N	N
hdsrfctl	Y	Y	N
hdssamqin	Y	Y	N
hdsshmclean	Y	Y	N

Command name	Information that is output		
	Command's arguments	Command's return value	Details in the event of an error
hdsstart	Y	Y	Y
hdsstate	N	N	N
hdsstop	Y	Y	Y
hdstrcredit	N	N	N
pdlogchg	N	N	N
pdls	N	N	N
pdrplstart	N	N	N
pdrplstop	N	N	N

Legend:

Y: Output

N: Not output

hdechgstatus (change the status of the source Datareplicator)

Function

The `hdechgstatus` command changes the maximum usage rate of the extraction information queue file. The information changed by this command is retained beyond termination or start of the source Datareplicator.

Format

```
hdechgstatus -c reset -k max_ratio
```

Options

`-c reset`

Specifies that the information specified in `-k` is to be reset.

`k max_ratio`

Specifies that the maximum usage rate for the extraction information queue file is to be changed.

Rules

- If the `hdechgstatus` command terminates normally, it returns 0. If it terminates abnormally, it returns 1. If the following conditions are satisfied, the command returns 11:
 1. There is a back-end server for which `true` has been specified in the `extsuppress` operand in the extraction environment definition.
 2. The back-end server noted in condition 1 above contains the table subject to extraction processing.
 3. The unit consists only of back-end servers that satisfy conditions 1 and 2.

Notes

- You can execute the `hdechgstatus` command only while the source Datareplicator is active. An error results if you execute it while the source Datareplicator is inactive.
- If any nodes are inactive or shut down at the time that the `hdechgstatus` command is executed, the command changes the status of only the active nodes. To change the status of nodes that were not changed, re-execute the command while all nodes are active.

hdeevent (issue an event at the source Datareplicator)

In the Windows edition of Datareplicator or HiRDB version 08-02 or later, execute this command from the source HiRDB's command prompt.

Function

The `hdeevent` command sends a specified event code to a target system in order to control import processing. Before you can use the `hdeevent` command, you must create an event control table at the source HiRDB. For details about the event control table, see *4.6.7 Designing the event control table*.

Format

```
hdeevent -n event-code
          [ -u user-ID/password ] [ -s target-identifier ]
```

Options

-n event-code

Specify an event code. The specifiable event codes are as follows:

0

Stops the transmission process.

1-255

Sends the specified event code to the target Datareplicator.

-u user-ID/password

user-ID ~ <symbolic name of 1-8 characters>

password ~ <symbolic name of 1-28 characters>

Specify the user ID and password used to create the event control table. Both user ID and password are case-sensitive. Do not specify any spaces before or after the delimiter slash (/). For the user ID, specify a symbolic name of 1 to 8 characters; for the password, specify a symbolic name of 1 to 28 characters.

If you omit the `-u` option, Datareplicator assumes the value of the `PDUSER` environment variable for the source HiRDB. If this environment variable is not specified, the `hdeevent` command results in an error during connection establishment processing with the source HiRDB.

-s target-identifier

Specify the identifier of the target that is to be the destination of the event.

Specifying `**` as the target identifier will result in an invalid-argument error. If you omit the `-s` option, Datareplicator sends the event to all destinations.

Rules

- When the `hdeevent` command is executed, the source Datareplicator issues an SQL statement to the event control table that has been created at the source HiRDB. The update log for this SQL statement is stored in the system log file. The source Datareplicator extracts this update information from the system log and sends it as an event to the target system(s).
- Specify the event code and import processing to be executed in the target Datareplicator's import environment definition. If an event is issued and its event code is not defined in the import environment definition, there is no effect on the import processing. However, a message is output to the target system's syslog file.
- If the `hdeevent` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.
- If the event code is outside the range of 1 to 255, the event will not be sent to the target Datareplicator.

Notes

- You can execute the `hdeevent` command only while the source HiRDB is active. An error results if you execute it while the source HiRDB is inactive.
- Execute the `hdeevent` command while no transactions are being processed at the source system. If it is executed during transaction processing, Datareplicator might not be able to identify the transaction to which the specified event applies. In such a case, conformity between the source and target databases might be lost.
- If the source HiRDB is a parallel server, execute the `hdeevent` command from the server machine containing the system manager. If the `hdeevent` command is executed from a server machine that does not contain the system manager and the `-s` option is specified, an error results and the `KFRB04001-E` message is output.
Even if audit trail collection is suppressed, Datareplicator ignores the `hirdb_audit_trail` operand and performs audit trail collection.

hdefcopy (copy the current source file)

Function

If a file duplexing error occurs in the source system, disabling one of the physical files (which is placed on `HOLD` status), the `hdefcopy` command copies from the normal file into the erroneous file.

This processing restores the erroneous file to usable status (`ACTIVE` status).

Format

```
hdefcopy -f source-physical-file-name -t target-physical-file-name
```

Options

`-f` *source-physical-file-name*

Specify the absolute path name of the source (normal) physical file.

`-t` *target-physical-file-name*

Specify the absolute path name of the target (erroneous) physical file.

Rules

- Execute the command for copying the current file at the node that contains the file that is to be restored.
- The current file copy command is executable when Datareplicator is inactive. To restore the data linkage file, HiRDB must also be inactive.
- A command error occurs if the source or target physical file name is not defined correctly in the duplexing definition.
- A command error occurs if the status of the source physical file is not `Active`.[#]
- A command error occurs if the status of the target physical file is not `Hold`.[#]
- If the file to be copied is a large file, the command might take a long time to execute. For details, see *6.11.2 Estimating the command execution time when large files are used*.

[#]

Use the `hdefstate` command to check the status of the physical file. For details, see the `hdefstate` command (display the status of duplexed source files).

hdefstate (display the status of duplexed source files)

Function

The `hdefstate` command displays the status of duplexed files (physical files) in the source system.

Format

`hdefstate [-f logical-file-name]`

Options

`-f logical-file-name`

Specify the absolute path name of the logical file whose status is to be displayed.

If this operand is omitted, the command displays the status of all duplex files at the server machine where this command is executed.

Output format

The `hdefstate` command displays the status of the extraction information queue files in the following format:

```

*****
**          HiRDB Datareplicator file status information          **
**          Sun Aug 25 12:58:52 2002                            **
*****
Logical file name = /users/repli/quefile001_bes1                ....1.

sys  status  type  Physical file name                ....2.
a   a-----  equ  /hd01/hde/qufile001_bes1_1
b   a-----  equ  /hd01/hde/qufile001_bes1_2
-----
Logical file name = /users/repli/quefile002_bes1

sys  status  type  Physical file name
a   a-----  equ  /hd01/hde/qufile002_bes1_1
b   h-----  equ  /hd01/hde/qufile002_bes1_2

Number of Logical file Information = 2                ....3.

```

1. Indicates the logical file name.
2. Indicates status information for the physical file.
 - `sys`: Physical file system
 - For the primary system, `a` is displayed; for the secondary system, `b` is displayed:

Output	Description
a	Primary file
b	Secondary file

status: Physical file status. The character in column 1 represents the status:

Output	Description
a	Current file status (ACTIVE)
h	Error status (HOLD)
*	Status cannot be obtained.

type: Type code for the physical file:

Output	Description
emt	Extraction master status file
sst	Server status file
rpl	Data linkage file
equ	Extraction information queue file
---	Other file
***	File type cannot be obtained

Physical file name: Name of the physical file.

- Indicates the number of logical file information items that have been displayed.

hdemodq (modify the organization of extraction information queue files)

Function

The `hdemodq` command is used offline to change the organization of the extraction information queue files. This command can execute the following processing:

- Register additional extraction information queue files.
- Release the registration of extraction information queue files that are no longer needed.
- Display offline information about the extraction information queue files.

Format

```
hdemodq [ -x host-name ]-b HiRDB-server-name
        { -l | -a extraction-information-queue-file-name
          [ -n additional-queue-duplexing-definition-file ]
          | -d extraction-information-queue-file-name }
```

Options

`-x host-name` ~ <symbolic name of 32 characters>

Specify the host name of the corresponding node.

Whether this option is required depends on the values of the `nodecontrol` and `errfile_unique` operands in the extraction system definition.

The following table shows the relationship between the `-x` option and the `nodecontrol` and `errfile_unique` operands in the extraction system definition:

nodecontrol and errfile_unique operand values	-x option
<ul style="list-style-type: none"> • nodecontrol=unit • errfile_unique=false 	Ignored, if specified.
<ul style="list-style-type: none"> • nodecontrol=server • errfile_unique=false 	
<ul style="list-style-type: none"> • nodecontrol=unit • errfile_unique=true 	Takes effect. Omitting the option results in an error.
<ul style="list-style-type: none"> • nodecontrol=server • errfile_unique=true 	Ignored, if specified.

`-b HiRDB-server-name` ~ <symbolic name of 8 characters>

Specify the HiRDB server name of the extraction environment that is to be processed by the `hdemodq` command. The command is executed in the environment of the source Datareplicator that corresponds to the specified server name to display information about the queue file, to register an additional file, or to release a registration. Note that you must also specify this option when the source HiRDB is a single server.

-l

Specify this option to display information about the extraction information queue file.

-a *extraction-information-queue-file-name*

To register an additional extraction information queue file, specify the absolute or relative path name of the extraction information queue file to be added. If a relative path name is specified, the command assumes `$HDEPATH/relative-path-name` as the absolute path.

The specified name must be unique in the source system. If the source HiRDB is a parallel server, specify a name that is unique within each back-end server.

- The actual name of the extraction information queue file is the specified name with `_server-name` appended. If you are specifying the absolute path name, make sure that the actual name including `_server-name` does not exceed 125 bytes. If you are specifying a relative path name, make sure that the actual name, which is `_server-name` appended to the assumed absolute path name, does not exceed 125 bytes.

The type of the queue file to be registered must be the same as the type of the existing queue files (in UNIX, if the existing queue files constitute a Datareplicator file system area, use a character special file).

If the file type is regular file for UNIX or Windows file, the `hdemodq` command creates the file.

- If the file type is a character special file for UNIX, the user must have created the file before executing the `hdemodq` command. The file name must be `_server-name` appended to the name specified in the `-a` option, and the file size must be equal to or greater than the size of the extraction information queue file.
- If this option is specified when large files are used, the command might take a long time to execute. For details, see *6.11.2 Estimating the command execution time when large files are used*.

-n

Specify this operand to add a duplexed extraction information queue file. Specify in this operand the absolute path name of the additional queue

duplexing definition file[#] that defines the organization of the extraction information queue file to be added.

When this operand is specified, the command treats the extraction information queue file name specified in `-a` as the logical file name.

#: The syntax for the additional queue duplexing definition file is the same as for the duplexing definition file. For details about the duplexing definition file, see *5.7 Duplexing definition (source)*.

If a duplexed queue file is added or deleted, the duplexing control file (see *3.9 Duplexing files*) is also updated as an extension of command processing. Therefore, if you employ a system switchover configuration, you must copy the duplexing control file to the secondary system after adding or deleting a queue. To keep this queue addition or deletion effective after the subsequent initialization of Datareplicator, you need to add or delete applicable duplexing definitions.

`-d extraction-information-queue-file-name`

Specify the absolute or relative path name of the extraction information queue file whose registration is to be released. If a relative path name is specified, the command assumes `$HDEPATH/relative-path-name` as the absolute path.

The actual name of an extraction information queue file is the specified name with `_server-name` appended.

Output format

The `hdemodq` command displays the status of the extraction information queue files in the following format:

```

hdeid = fa                                     ... 1.
server name = pbes01                           ... 2.
wrap count = 0                                  ... 3.
status          size create date              file path    ... 4.
qufile001 u----- 8388000 Fri Sep 2 14:25:09 2005 /.../extque01_pbes1
qufile002 u----- 8388000 Fri Sep 2 14:25:09 2005 /.../extque02_pbes1
qufile003 e----- 8388000 Fri Sep 2 14:25:09 2005 /.../extque03_pbes1
    
```

1. Indicates the source Datareplicator identifier.
2. Indicates the server name of the source HiRDB.
3. Displays the wrap-around count for the extraction information queue file. Once the wrap count reaches 4294967295, it is reset to 0.
4. Indicates information about the extraction information queue files.
 status: Displays the status of extraction information queue files. The following

information is displayed:

No.	Code location	Description	Value	Status
1	Col. 1	Whether there is any untransmitted extraction information	u	There is extraction information waiting to be sent to the target.
2			e	There is no more extraction information waiting to be sent to the target.
3	Cols. 2-7	Undefined	--	--

`size`: Displays the size of the extraction information queue file in kilobytes.

`create date`: Displays the date and time the extraction information queue file was created.

`file path`: Displays the path of the extraction information queue file.

Note

If an error that indicates a nonexistent extraction node master error information file occurs, check and, if necessary, revise the following:

Check item	Action to be taken if necessary
Whether the extraction node master error information file has been deleted illegally.	Re-execute this command with the following procedure: <ol style="list-style-type: none"> 1. Restart the source Datareplicator. 2. Terminates the source Datareplicator. 3. Re-execute the <code>hdemodq</code> command.
Whether specification of the <code>-x</code> option is correct, if specified.	Correct the specification.

hdeprep (create an extraction definition preprocessing file)

In the Windows edition of Datareplicator or HiRDB version 08-02 or later, execute this command from the source HiRDB's command prompt.

Function

The `hdeprep` command analyzes the contents of the extraction definition file, converts it to the internal format, and creates an extraction definition preprocessing file.

Format

```
hdeprep -f extraction-definition-filename [ -u user-ID/password ]
          [ -k {not_null_unique|unique|none} ]
```

Options

`-f extraction-definition-filename`

~ <pathname of 1-64 bytes>

Specify the name of the extraction definition file, as an absolute or relative pathname. If you specify a relative pathname, Datareplicator assumes `$(HDEPATH)/relative-pathname` as the absolute pathname. In this case, specify the pathname so that the final absolute pathname will not exceed 125 bytes.

`-u user-ID/password`

user-ID ~ <symbolic name of up to 8 characters>

password ~ <symbolic name of up to 28 characters>

Specify the user ID and password of a user who has the DBA privilege for the source Datareplicator. Both user ID and password are case-sensitive. Do not specify any spaces before or after the delimiter slash (/). For the user ID, specify a symbolic name of 1 to 8 characters; for the password, specify a symbolic name of 1 to 28 characters.

If you omit the `-u` option, Datareplicator assumes the value of the `PDUSER` environment variable for the source HiRDB. If this environment variable is not specified, the `hdeprep` command results in an error during connection establishment processing with the source HiRDB.

`-k {not_null_unique|unique|none}`

Specify the condition for unique check on the mapping key column.

This value becomes the default when the `check` clause is omitted in the extraction definition. The specification values and the details of checking are the same as for the `check` clause of the extraction definition. For details, see the `check` clause

under *Explanation of the operands* in 5.5 *Extraction definition*.

If the specified condition is not satisfied, the command terminates abnormally. When this option is omitted, `not_null_unique` is assumed.

Rules

- Execute the `hdeprep` command after executing the `hdestart -i` command.
- You can execute the `hdeprep` command only while the source HiRDB is active. Execute the `hdeprep` command while the source Datareplicator is inactive. You can execute the command while the source Datareplicator is active, but when you do so the command's execution results do not become effective at the source Datareplicator until the next time the source Datareplicator is started (the execution results do not take effect at the source Datareplicator during the session in which the command is executed).
- Datareplicator creates the extraction definition preprocessing file as `hde_prpfile` under `$HDEPATH`.
- If the `hdeprep` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.

Notes

- You must execute the `hdeprep` command in the following cases:
 - The `hdestart -i` command has been executed.
 - The definition of a table subject to extraction processing has been modified at the source HiRDB.
 - The source Datareplicator's extraction definition has been modified.
 - An event control table has been created.
- During analysis of the extraction definition, Datareplicator assumes the character code system specified in the `dblocale` operand.

hdeshmclean (delete the source Datareplicator's shared resources)

Function

The `hdeshmclean` command deletes shared resources (processes, shared memory, semaphores) that remain when the source Datareplicator terminates abnormally. This command is only applicable to the UNIX edition. The Datareplicator administrator executes this command.

Format

```
hdeshmclean [ -x host-name ]
             [ -l clean[ -q{resp|noresp} ] [ -w ] [ -t{mst|nmt|both} ] ]
```

Options

`-x host-name` ~ <identifier of 1-32 characters>

Specify the name of the host that contains the shared resources to be deleted. This option must be specified in the following cases:

Condition	Value to be specified in this option
<code>true</code> is specified in the <code>errfile_unique</code> operand in the extraction system definition	Specify the name of the host that contains the shared resources to be deleted (specify the name that matches the <code>xxxx</code> part of the file <code>errfile1_xxxx</code> that has been created under <code>\$HDEPATH</code> on the corresponding server). Note that there is no need to specify this option when the command is executed with the <code>-t mst</code> option specified on the server where only the extraction master process is running.
<code>server</code> is specified in the <code>nodecontrol</code> operand in the extraction system definition	Specify the name of the back-end server that contains the shared resources to be deleted (specify the name that matches the <code>xxxx</code> part of the file <code>errfile1_xxxx</code> that has been created under <code>\$HDEPATH</code> on the corresponding back-end server).

`-l clean`

Specify this option to delete shared resources. If this option is omitted, the command only displays the remaining shared resources.

`-q{resp|noresp}`

Specify whether the `KFRB04331-Q` message is to be displayed for confirmation before the shared resources are deleted.

`resp`

Outputs the KFRB04331-Q message. The command deletes the shared resources only when *y* is entered.

`noresp`

Deletes the shared resources without displaying the KFRB04331-Q message.

`-w`

Specify this option to delay termination of this command until deletion of the shared resources has been completed. If this option is omitted, whether deletion of the shared resources has been completed when the command terminates cannot be guaranteed.

`-t {mst | nmt | both}`

Specify the resources that are to be deleted when this command is executed on the server where both types of resources exist.

`mst`

Deletes resources managed by the extraction master process.

`nmt`

Deletes resources managed by the extraction node master process.

`both`

Deletes resources managed by both the extraction master process and the extraction node master process.

Output format

`nodemst` specified in the `sendcontrol` operand in the extraction system definition

<<<*** Common information ***>>>	1.
Shmid : statinf = 13467653	2.
Semid : mstcmd = 0	3.
PID : hdemaster = 7241	4.
PID : hdstrcrv = 25754	5.
<<<*** Node information (drum2) ***>>>	6.
Shmid : extcam = 13500422	7.
extdef = 13533191	8.
errtxt = 13565960	9.
rmtcmd = 0	10.
Semid : extcam = 8912925	11.
rmtcmd = 0	12.
PID : hdenodenst = 7244	13.
PID : hdstrcrv = 25757	14.
<<<*** Server information (bes1) ***>>>	15.
Shmid : tminf(ticb) = 0	16.
tminf(excb) = 0	17.
tminf(d2cb) = 0	18.
sndprcinf = 0	19.
Semid : svrcam = 8945694	20.
PID : hdecapture = 7247	21.
PID : hdesender(SND01) = 7248	22.
<<<*** Server information (bes2) ***>>>	
Shmid : tminf(ticb) = 0	
tminf(excb) = 0	
tminf(d2cb) = 0	
sndprcinf = 0	
Semid : svrcam = 8945698	
PID : hdecapture = 7252	
PID : hdesender(SND01) = 7253	
⋮	

sendmst specified in the sendcontrol operand in the extraction system definition

```

<<<*** Common information ***>>> ..... 1.
Shmid : statinf          = 13467653 ..... 2.
Semid : mstcmd          = 0 ..... 3.
PID   : hdemaster       = 7241 ..... 4.
PID   : hdstrcrv        = 25754 ..... 5.

<<<*** Node information (drum2) ***>>> ..... 6.
  Shmid : extcom         = 13500422 ..... 7.
         extdef          = 13533191 ..... 8.
         errtxt          = 13565960 ..... 9.
         rmtcmd          = 0 ..... 10.
  Semid : extcom         = 8912925 ..... 11.
         rmtcmd          = 0 ..... 12.
  PID   : hdenodemst    = 7244 ..... 13.
  PID   : hdstrcrv      = 25757 ..... 14.

<<<*** Server information (bes1) ***>>> ..... 15.
  Shmid : tminf (ticb)   = 2333 ..... 16.
         tminf (excb)   = 3432 ..... 17.
         tminf (d2cb)   = 4522 ..... 18.
         sndprcinf      = 1123 ..... 19.
  Semid : svrcom         = 8945694 ..... 20.
  PID   : hdecapture    = 7247 ..... 21.
  PID   : hdesendmst    = 7248 ..... 22.
  PID   : hdesndprc     = 7249 .....
  PID   : hdesndprc     = 7250 ..... } ..... 23.
  PID   : hdesndprc     = 7251 .....

<<<*** Server information (bes2) ***>>>
  Shmid : tminf (ticb)   = 344
         tminf (excb)   = 566
         tminf (d2cb)   = 879
         sndprcinf      = 112
  Semid : svrcom         = 8945698
  PID   : hdecapture    = 7252
  PID   : hdesendmst    = 7253

      :
      :
  
```

Explanation of the output information

No.	Output information	Remarks
1	Header for the resources managed by the extraction master process	If there are no resources managed by the extraction master process, Nos. 1 through 5 are not displayed.
2	Shared memory ID for command communication	--
3	Semaphore IDs for locking status commands	--

No.	Output information	Remarks
4	Process ID of extraction master process	--
5	Process ID for activity trace collection process	--
6	Header for the resources managed by the extraction node master process	If there are no resources managed by the extraction node master process, items beginning with No. 6 are not displayed.
7	Shared memory ID for process-to-process communication	--
8	Shared memory ID for storing extraction definition	--
9	Shared memory ID for storing message text	--
10	Shared memory ID for command communication	If <code>server</code> is not specified in the <code>nodecontrol</code> operand in the extraction system definition, 0 is output.
11	Semaphore for locking nodes	--
12	Semaphore IDs for locking commands	If <code>server</code> is not specified in the <code>nodecontrol</code> operand in the extraction system definition, 0 is output.
13	Process ID of extraction node master process	--
14	Process ID of activity trace collection process	--
15	Resource header for each server	--
16	Shared memory ID for managing transaction information	If <code>server</code> is not specified in the <code>nodecontrol</code> operand in the extraction system definition, 0 is output.
17	Shared memory ID for managing update information in transaction	
18	Shared memory ID for managing branch information in transaction	
19	Shared memory ID for managing transmission process	
20	Semaphores for locking servers	
21	Process ID of extraction process	--
22	Process ID of transmission process	The transmission process name is displayed as follows: When <code>nodemst</code> is specified in the <code>sendcontrol</code> operand in the extraction system definition: <code>hdesender (target-ID)</code> When <code>sendmst</code> is specified in the <code>sendcontrol</code> operand in the extraction system definition: <code>hdesendmst</code>

No.	Output information	Remarks
23	Process ID of transmission process (<code>hdesndprc</code>)	This information is output when <code>sendmst</code> is specified in the <code>sendcontrol</code> operand in the extraction system definition. As many process IDs as there are active processes are displayed. If no transmission process (<code>hdesndprc</code>) is running, this information is not displayed.

Legend:

--: Not applicable

Rules

If the source Datareplicator cannot be started because there are shared resources remaining, execute the `hdestop` command to delete the shared resources. Use this command if the shared resources cannot be deleted by executing the `hdestop` command.

hdestart (start the source Datareplicator)

Function

The `hdestart` command starts the source Datareplicator in accordance with its extraction system definition. For details about the extraction system definition, see 5.2 *Extraction system definition*.

Format

```
hdestart {[ -i[ init ] [ -S target-identifier-number ] |
          [ -e ] [ -r ] [ -s[ target-identifier-number
                        [ { { ,target-identifier-number } } ... ] ] [ -L [ -H ] ] ] |
          [ -R -k queue -b HiRDB-server-name ] |
          [ -v [ -e ] [ -r ] ] }
```

Options

`-i [init] [-S target-identifier-number]`

Specify this option to initialize the source Datareplicator environment. When the `-i` option is specified, the source Datareplicator does not start.

When the `-s` option is specified, Datareplicator initializes only the transmission environment for the specified destination. When the `-s` option is omitted, Datareplicator initializes the following files and the entire source Datareplicator environment (when the `-s` option is specified, Datareplicator does not initialize files):

- Extraction information queue files
- Extraction master status file
- Extraction server status file
- Extraction master error information files
- Extraction node master error information files
- Extraction master trace files
- Extraction node master trace files
- Data linkage file

`init`

Specify this option to create all extraction information queue files specified in the extraction environment definition based on the size specified in the `queuesize` operand in the extraction environment definition. Depending on the specified file size, the creation processing might not be completed within the communication wait time specified in the `cmwaittime` operand in the

extraction system definition, resulting in an error (KFRB00607-E message). In such a case, increase the communication wait time and re-execute the initialization processing.

This option is ignored when the `-S` option is specified.

If this option is specified when large files are used, the command might take a long time to execute. For details, see *6.11.2 Estimating the command execution time when large files are used*.

`-S target-identifier-number`

Initializes the transmission environment for the specified target only (partial initialization). Datareplicator does nothing for targets that are not specified.

Specify the target identifier number, which is the numeric digits in `xx` or `xxxx` of the target identifier specified in the extraction system definition (`sendidxx` or `sendidxxxx`). The specified target identifier number must have been defined when the most recent initial start was executed. You cannot specify a newly added target identifier number. The partial initialization specified in this operand is applicable to transmission environment definitions specified with `sendidxx` (or `sendidxxxx`), or `senddefxx` (or `senddefxxxx`) in the extraction system definition.

The next time a transmission process with the specified target identifier is started after partial initialization, it will start sending the update information that has been written into the extraction queue by the extraction process. Datareplicator handles any change to a definition other than partial initialization in the same manner as a normal start.

Do not change a definition other than the target identifier that is subject to partial initialization. To perform partial initialization on multiple target identifiers, first change the definition of a target identifier, and then perform partial initialization; then, repeat this procedure for each additional target identifier.

`-e`

Specify this option to start only extraction processing at startup without starting transmission processing. If extraction processing was not started during the preceding execution of `hdestart`, you use this option to start extraction processing also.

`-r`

Specify this option to reset the data transmission count when the source Datareplicator is started.

This option is applicable when `false` is specified in the `send_counter_reset` operand in the extraction system definition.

hdestart (start the source Datareplicator)

-s [*target-identifier* [{ { , *target-identifier* } } . . .]] [-L [-H]]

Specify this option to start only transmission processing or update-SQL output processing at startup without starting extraction processing. If transmission processing was not started during the preceding execution of `hdestart`, you use this option to start transmission processing also.

target-identifier

Specify a target identifier specified in the extraction system definition in order to start only transmission processing or update-SQL output processing for the specified target. If you specify multiple target identifiers, do not specify any spaces before or after a delimiter comma. Specifying `**` as a target identifier will result in an invalid-argument error.

If you do not specify any target identifiers, Datareplicator starts transmission processing or update-SQL output processing for all target identifiers.

-L

Start update-SQL output processing at startup.

-H

This option is applicable only when the `-L` option is specified.

Specify this option to output the values of CHAR, VARCHAR, MCHAR, MVARCHAR, NCHAR, and NVARCHAR columns in hexadecimal during update-SQL output processing. For details about the format and contents of the column values that will be output, see *9.8(2) Update-SQL file*.

Specify this option if the column values output to the update-SQL file cannot be read as character data. Column values cannot be read as character data if they contain characters that cannot be displayed, such as control codes (0x00 through 0x1F) and hexadecimal numbers.

-R

Specify this option to recover the extraction definition.

-k *queue*

Recovers the extraction information queue file by using the facility for recovering the extraction information queue file. For details about the facility for recovering the extraction information queue file, see *9.7 Facility for recovering the extraction information queue file*.

-b *HiRDB-server-name* ~ <symbolic name of 8 characters>

Specifies the HiRDB server name of the extraction environment that is to be recovered.

If the source HiRDB is a single server, specify the server name of the single

server (server name that is specified in the HiRDB's `pdstart -s` command).

If the source HiRDB is a parallel server, specify the name of the back-end server that is to be recovered. This must be the server name specified in the extraction environment definition for the back-end server. If a wrong server name is specified, the command outputs the `KFRB00711-E` message, and then terminates with an error.

Specify only one server name. If there are multiple servers subject to recover, execute the command for each server subject to recovery.

-v

Specify this option to perform data linkage recovery via the system log file.

For details about data linkage recovery via the system log file, see *9.5 Data linkage recovery via the system log file*.

Rules

- If the `hdestart` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.
- The start mode depends on the previous termination mode. The following table shows the relationship between the start mode and the previous termination mode.

Table 7-3: Relationship between the source Datareplicator's start and termination modes

Start mode	Previous termination mode	Action at startup
Initial start	N/A [#]	At initial start, the source Datareplicator initializes files, as shown in <i>Table 7-4 Handling of each file when the -i option is specified (-S option is omitted)</i> ; the source Datareplicator does not start.
Partial initial start	N/A [#]	At partial initial start, the source Datareplicator initializes the transmission environment only for a specified target; the source Datareplicator does not start.
Normal start	Normal termination	The source Datareplicator starts in accordance with its definitions without inheriting the settings used at the previous session.
Restart	Abnormal termination or forced termination	The source Datareplicator starts with the settings used at the previous session in order to guarantee conformity between the source and target databases.

[#]: Executing the `hdestart` command with the `-i` option specified results in initial start or partial initial start, regardless of the previous termination mode.

- The following table shows the handling of each file when the `-i` option is

hdestart (start the source Datareplicator)

specified (-s option is omitted).

Table 7-4: Handling of each file when the -i option is specified (-S option is omitted)

File type	No existing file	Type of existing file		Datareplicator file system area ^{#1}	
		OS regular file	Character special file ^{#1}	In initialized status	Being used
Extraction information queue file	Created in OS regular file format	Re-created	Does nothing ^{#2}	File is allocated.	File is reallocated.
Extraction master status file			Header is initialized. ^{#2}	N/A	N/A
Extraction server status file			Header is initialized. ^{#2}	File is allocated.	File is reallocated.
Extraction master error information file			Cannot be created with a character special file.	N/A	N/A
Extraction node master error information file				N/A	N/A
Extraction master trace file				N/A	N/A
Extraction node master trace file				N/A	N/A
Data linkage file			Header is initialized. ^{#2}	File is allocated.	File is reallocated.

Legend:

N/A: Cannot be stored in the Datareplicator file system area.

#1

UNIX Datareplicator supports character special files and Datareplicator file system areas.

Windows Datareplicator executes the processing described in the *OS regular file* column.

#2

Use an OS command to re-create the file.

- If the `hdestart` command is executed with the `-i` option omitted, the source Datareplicator's processes start according to the specified options. For the relationship between the options and the processes that are started, see Table 7-7 *hdestart commands to be executed*. If the source Datareplicator is already running and the `hdestart` command is executed with the `-i` option omitted, only the inactive processes are started among all processes corresponding to the specified options.
- Depending on the status of the source Datareplicator, you might not be able to execute the `hdestart` command. The following table shows the relationship between the source Datareplicator's status and when the `hdestart` command can be executed

Table 7-5: Relationship between the source Datareplicator's status and when the `hdestart` command can be executed

Source Datareplicator's status	Specified option						
	None	-i	-e	-s	-s -L	-R	-v
Starting	--	--	--	--	--	--	--
Active [#]	Y	--	Y	Y	--	--	--
During update-SQL output	--	--	--	--	Y	--	--
Stopping	--	--	--	--	--	--	--
Inactive	Y	Y	Y	Y	Y	Y	Y

Y: Can be executed.

--: Cannot be executed (results in an error).

#

If `hdestart` other than `hdestart -I` is executed while extraction or transmission processing is stopped, the extraction and transmission processing are restarted from the point where they had stopped.

- You can combine multiple options when you execute the `hdestart` command. The following table shows the combinations of `hdestart` command options.

Table 7-6: Combinations of `hdestart` command options

Command option	-i	-e	-s	-s -L	-r	-R	-k	-v
-i	--	--	--	--	--	--	--	--
-e	--	--	Y	--	Y	--	--	Y
-s	--	Y	--	--	Y	--	--	--

hdestart (start the source Datareplicator)

Command option	-i	-e	-s	-s -L	-r	-R	-k	-v
-s -L	--	--	--	--	Y	--	--	--
-r	--	Y	Y	Y	--	--	--	Y
-R	--	--	--	--	--	--	Y	--
-k	--	--	--	--	--	Y	--	--
-v	--	Y	--	--	Y	--	--	--

Y: Executable combination.

--: Not an executable combination (results in an error).

- If some source processes stop due to an error, execute the same command as during startup. This will restart only those source processes that are stopped due to the error, without affecting the processes that are running normally.
- Figure 7-1 *hdestart command action when the source HiRDB is a parallel server* shows the `hdestart` command action when the source HiRDB is a parallel server, and Table 7-7 *hdestart commands to be executed* lists the `hdestart` commands to be executed. The numbers in Figure 7-1 *hdestart command action when the source HiRDB is a parallel server* correspond to the numbers in Table 7-7 *hdestart commands to be executed*.

Figure 7-1: hdestart command action when the source HiRDB is a parallel server

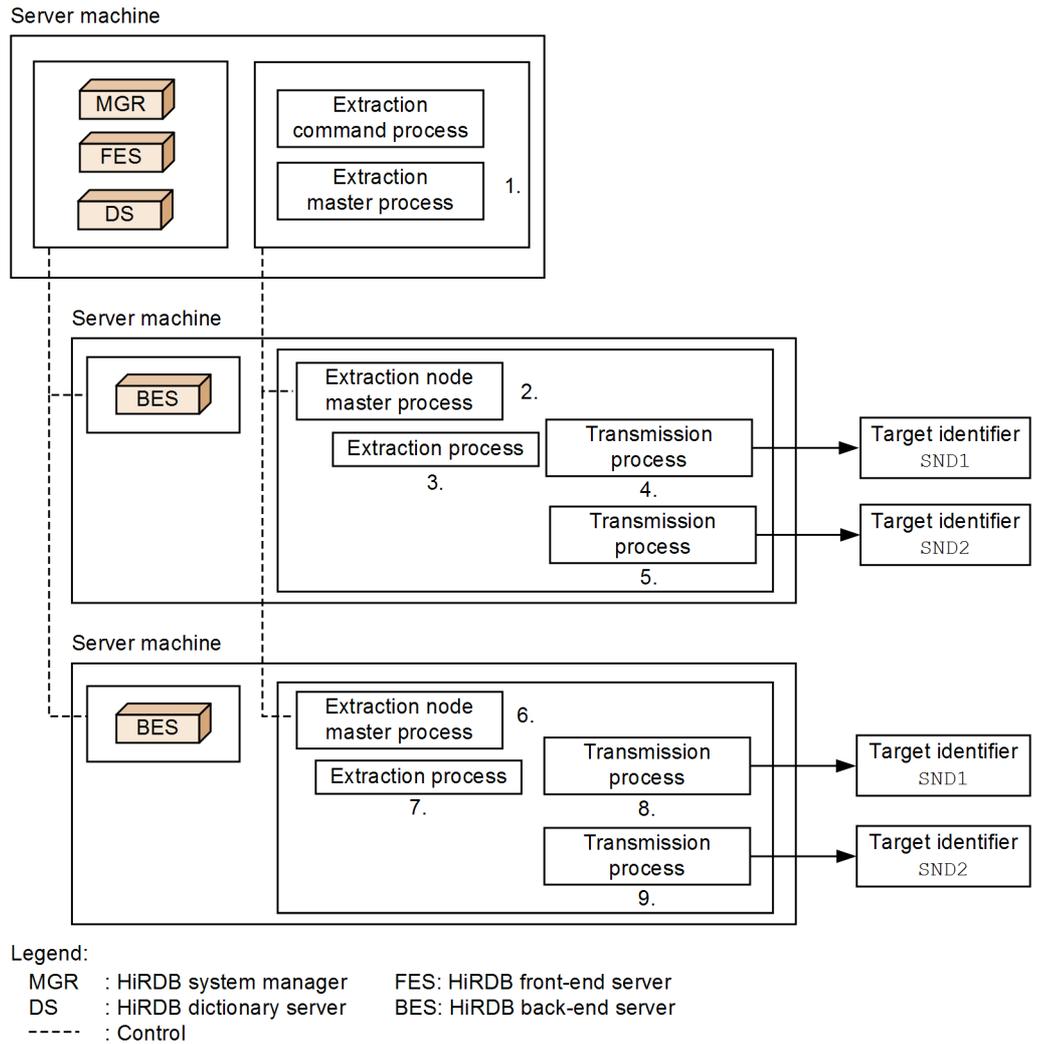


Table 7-7: hdestart commands to be executed

Command to be executed	Action
hdestart -i	Initialize the source Datareplicator environment
hdestart	Start all processes (1 to 9)

hdestart (start the source Datareplicator)

Command to be executed	Action
<code>hdestart -e</code>	Start the extraction master process (1), the extraction node master processes (2 and 6), and the extraction processes (3 and 7).
<code>hdestart -s</code>	Start the extraction master process (1), the extraction node master processes (2 and 6), and all transmission processes [#] (4, 5, 8, and 9)
<code>hdestart -s SND1</code>	Start the extraction master process (1), the extraction node master processes (2 and 6), and the transmission processes [#] whose target identifier is SND1 (4 and 8)
<code>hdestart -e -s SND1</code>	Start the extraction master process (1), the extraction node master processes (2 and 6), the extraction processes (3 and 7), and the transmission processes [#] whose target identifier is SND1 (4 and 8)

#

If `sendmst` is specified in the `sendcontrol` operand, the transmission master process and the transmission processes are started.

Notes

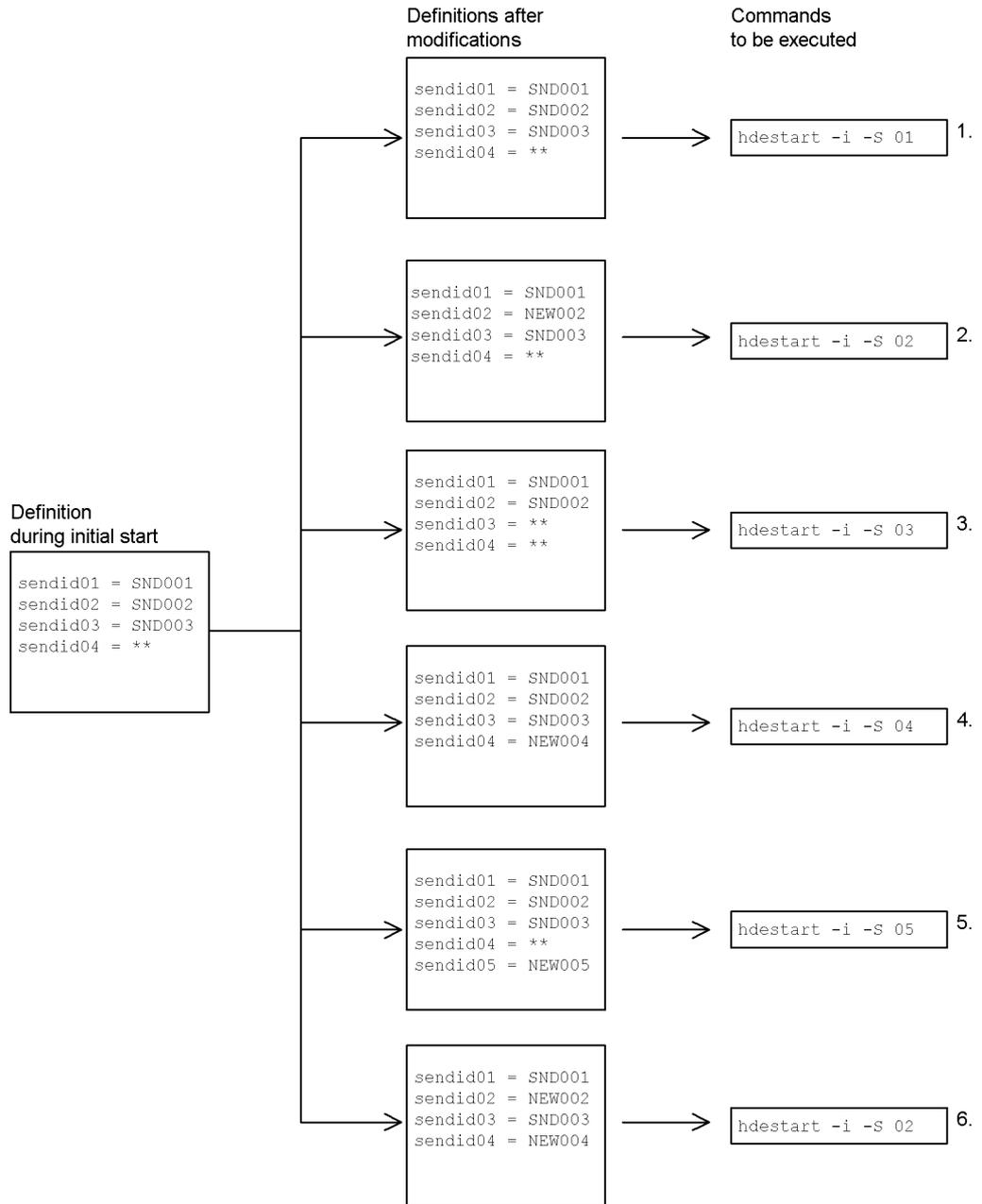
- If you specify neither the `-e` nor the `-s` option, the source Datareplicator executes extraction processing, and then immediately executes transmission processing.
- Before you can use the `hdestart` command, you must specify the `/etc/inetd.conf` communications environment variables. For details about the communications environment setup procedure, see *2.4 Specifying environment variables (UNIX)* or *2.8 Specifying environment variables (Windows)*.
- You must execute the `hdeprep` command before you can execute the `hdestart` command in the following cases:
 - Data linkage is to be started for the first time.
 - The `hdestart -i` command has been executed.
 - A source table definition or the extraction definition is to be modified.
- The user executing the `hdestart` command must belong to the same user group as the source HiRDB executor.
- Specify the `hdestart -i` command while the source HiRDB is not executing HiRDB Datareplicator linkage. Use the `pdls` command to determine whether the source HiRDB is executing HiRDB Datareplicator linkage. If HiRDB Datareplicator linkage is executing, use the `pdrplstop` command to cancel it.
- If an error occurs during partial initial start, you cannot execute normal startup, extraction definition preprocessing, or another partial initial start with a different target identifier number while the error remains uncorrected. To correct a partial

initial start error, eliminate the cause of the error, re-execute partial initial start with the same target identifier number specified, and then terminate Datareplicator normally; or, execute initial start on the entire extraction environment.

- When you execute the `hdestart` command with the `-i` option specified (except in the case of partial initial start), you must enter a reply to confirm that you want processing to start.
- You can specify partial initial start only for target identifier numbers that have been defined at the time of the most recent initial start. You cannot specify a new target identifier number that has been added since then. If the number of targets is expected to increase, specify `**` for future target identifiers.

The following is a specification example of partial initial start:

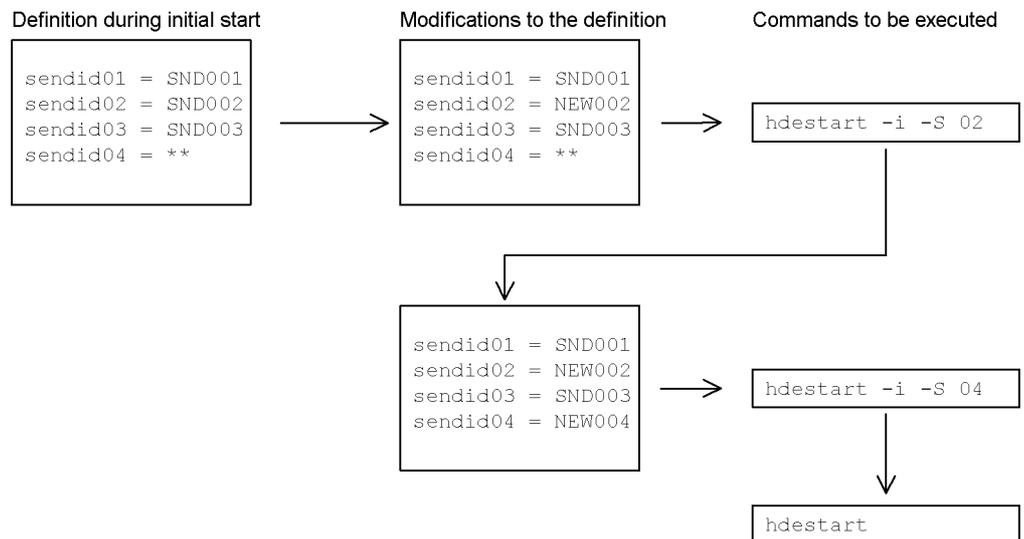
hdestart (start the source Datareplicator)



1. Executes partial initial start for target identifier number 01 (target identifier SND001).

2. Executes partial initial start for target identifier number 02 (target identifier NEW002) after modifying the definition.
3. Executes partial initial start for target identifier number 03 (target identifier **), which became an absent number after the definition was modified.
4. Executes partial initial start for target identifier number 04 (target identifier NEW004), which has been newly defined.
5. Results in a command error because target identifier number 05 (target identifier NEW005), which was added by modifying the definition, had not been defined at the time of the initial start.
6. Results in a command error because target identifier number 004 (target identifier NEW004) was modified, which is not the specified target identifier number (target identifier NEW002).

To execute partial initial start for more than one target identifier, modify the definition for one target identifier at a time, and then execute partial initial start. After all partial initializations have been completed, execute normal start. The following is an example of the execution procedure:



- The following is the execution procedure for partial initial start:

1. Check that the source Datareplicator is stopped.
2. Use a text editor to modify the definitions.
3. Specify the target identifier number subject to initialization in the `hdestart -i -S` command in order to execute partial initial start.

hdestart (start the source Datareplicator)

4. Execute the `hdeprep` command.
5. Execute normal start on the source Datareplicator.
 - If you use a system switchover configuration, you must activate the resources that are used by HiRDB Datareplicator (shared disk unit and network) prior to initialization or startup.

hdestart_n (partially start the source Datareplicator)

Function

The `hdestart_n` command partially starts the source Datareplicator.

The following table provides an overview of this command's operation:

-b option	Process to be started			
	Extraction master process	Extraction node master process	Extraction process	Transmission process
Omitted	Y#1, #2	--	--	--
Specified	--	Y#1, #3, #4	Y#4	Y#4

#1: If the process is already active, the command outputs a warning message and terminates normally.

#2: If all extraction node master processes are in inactive status when the wait time (`-t` option value) since the extraction master process started elapses, the extraction master process terminates automatically.

#3: Once started, the extraction node master process waits until it can communicate with the extraction master process. If it cannot communicate with the extraction master process within the specified wait time (`-t` option value), the extraction node master process terminates automatically.

#4: The processes to be started are the extraction node master process that corresponds to the back-end server specified in the `-b` option, and the extraction and transmission processes under it. The command has no effect on processes in other back-end servers.

Format

```
hdestart_n [ -b server-name -x extraction-master-process's-host-name
            -n service-name [ -c initial|continue ] [ -r ] ]
            [ -t wait-time ]
```

Options

```
-b server-name -x extraction-master-process's-host-name -n service-name [ -c
initial|continue ]
```

Specifies that the extraction node master process at the source Datareplicator is to be started. When this option is omitted, the extraction master process is started.

`-b server-name`

~ <identifier of 1-8 characters>

hdestart_n (partially start the source Datareplicator)

Specifies the name of the back-end server that corresponds to the extraction node master process to be started.

`-x extraction-master-process's-host-name`

~ <identifier of 1-32 characters>

Specifies the name of the host where the extraction master process is run (host with IP address inheritance that has been allocated for the source Datareplicator) in order to issue a start request to the extraction master process. This must be a host name that was specified in the `connection_accept_hostname` operand in the extraction system definition.

If the specified host name is invalid, the `hdestart_n` command terminates normally, but the extraction node master process that was started by this command will terminate with an error due to a communication timeout.

`-n service-name`

~ <identifier of 1-64 characters>

Specifies the service name used to establish communication between the extraction master process and extraction node master process. This service name must have been specified in the `connection_accept_service` operand in the extraction system definition.

The following table describes what occurs when the wrong service name is specified:

-n option value	-x option value	Operation
Service name specified in the <code>mstservice</code> operand in the extraction system definition	Name of the host at which the source Datareplicator is located	The extraction node master process starts at the host specified in the <code>-x</code> option and the host where the <code>hdestart_n</code> command was executed, and this command terminates normally. After that, both extraction node master processes terminate with a communication error.
Service name specified in the <code>hdeservice</code> operand in the transmission environment definition	Name of the host at which the target Datareplicator is running	The <code>hdestart_n</code> command terminates normally, and then the extraction node master process terminates with a communication timeout error. The target Datareplicator detects an error during port checking (detail code = 16) and refuses reception.
Other		The <code>hdestart_n</code> command terminates normally, and then the extraction node master process terminates with a communication timeout error.

`-c initial | continue`

Specifies that only the extraction or transmission process that had been activated by a command request when the previous source Datareplicator

was terminated is to be started.

The following describes the operation of extraction and transmission processes depending on the specification of the `-c` option.

Process	Previous termination status	-c option value	
		initial	continue
Extraction process	continue was specified in the <code>-c</code> option of the <code>hdestop_n</code> command	Started	Started
	Error termination		Started ^{#1}
	Abnormal termination		
	Other		Not started
Transmission process	continue was specified in the <code>-c</code> option of the <code>hdestop_n</code> command	Started	Started
	Error termination		Started ^{#1}
	Abnormal termination		
	Other		Not started
Update-SQL output process	--	Not started ^{#2}	Not started ^{#2}

#1: With respect to error termination and abnormal termination, taking into account the case where an error results during system switchover processing, the process is started regardless of the `-c` option's specification.

#2: The update-SQL output process is not started and the extraction and transmission processes are started.

`-r`

Specify this option to reset the data transmission count when the source Datareplicator is started.

This option is applicable when `false` is specified in the `send_counter_reset` operand in the extraction system definition.

`-t wait-time`

~ <unsigned integer> ((1-3600)) <<300>> (seconds)

Specifies the wait time until the first connection is established. This option depends on whether the `-b` option is specified.

The following table describes what the `-t` option's value represents, depending

on whether the -b option is specified:

-b option	Meaning of the -t option's value
Specified	Specifies the wait time until the extraction node master process started by the hdestart_n command establishes connection with the extraction master process. If connection cannot be established within the specified amount of time, the extraction node master process terminates with a timeout.
Omitted	Specifies the wait time until the extraction master process started by the hdestart_n command establishes connection with an extraction node master process. If connection cannot be established within the specified amount of time, the extraction master process terminates with a timeout.

This operand's value is applicable only to the first connection after the process has started. If another process is terminated after connection has been established, connection establishment is retried, but this retry is based on the specification of the connection_retry_time operand in the extraction system definition.

Rules

- To execute the hdestart_n command, you must specify server in the nodecontrol operand in the extraction system definition. If unit is specified in the nodecontrol operand, the operation is as follows:
 - When the -b option is specified

The hdestart_n command terminates normally, but the started extraction node master process terminates with a timeout. In such a case, you can use the hdestop_n command to terminate the extraction node master process without having to wait for the timeout to occur.
 - When the -b option is omitted

The hdestart_n command terminates normally, but the extraction master process terminates with a definition analysis error.
- If the hdestart_n command terminates normally, it returns 0. If it terminates abnormally, it returns 1.
- Table 7-9 lists and describes the -b option value and the operation after execution of the hdestart_n command.

Table 7-8: -b option value and operation after execution of the hdestart_n command

Machine where command is executed		-b option value	Operation after execution of the hdestart_n command
System manager unit	Current	Omitted	The command starts only the extraction master process, and then terminates itself normally.
		The specified server does not exist in the source HiRDB.	The command starts the extraction node master process, and then terminates itself normally. The extraction master process outputs a request acceptance error (invalid server name) message, and the extraction node master process terminates with an error.
		The specified server exists in the source HiRDB but not in the system manager unit.	The command starts the extraction node master process, and then terminates itself normally. The corresponding extraction node master process starts, and then terminates with an input/output error on the extraction server status file.
		The specified server exists in the source HiRDB and in the system manager unit.	The command starts only the extraction node master process that corresponds to the specified server, and then terminates itself normally. The extraction node master process starts the extraction and transmission processes.
	Standby	Omitted	The command terminates normally. The extraction master process terminates with an input/output error on the extraction master status file.
		The specified server does not exist in the source HiRDB.	The command starts the extraction node master process, and then terminates itself normally. The extraction master process outputs a request acceptance error (invalid server name) message, and the extraction node master process terminates with an error.
		The specified server exists in the source HiRDB but not in the system manager unit.	The command starts the extraction node master process, and then terminates itself normally. The corresponding extraction node master process starts, and then terminates with an input/output error on the extraction server status file.
		The specified server exists in the source HiRDB and in the system manager unit.	

Machine where command is executed		-b option value	Operation after execution of the hdestart_n command
Non-system manager unit	Current	Omitted	The command terminates with an error because the extraction system definition file cannot be found.
		The specified server does not exist in the source HiRDB.	The command starts the extraction node master process, and then terminates itself normally. The extraction master process outputs a request acceptance error (invalid server name) message, and the extraction node master process terminates with an error.
		The specified server exists in the source HiRDB but not in the system manager unit.	The command starts the extraction node master process, and then terminates itself normally. When the extraction master process starts and issues a start request to the corresponding extraction node master process, the extraction node master process terminates with an input/output error on the extraction server status file.
		The specified server exists in the source HiRDB and in the system manager unit.	The command starts only the extraction node master process that corresponds to the specified server, and then terminates itself normally. The extraction node master process starts the extraction and transmission processes.
	Standby	Omitted	The command terminates with an error because the extraction system definition file cannot be found.
		The specified server does not exist in the source HiRDB.	The command starts the extraction node master process, and then terminates itself normally. The extraction master process outputs a request acceptance error (invalid server name) message, and the extraction node master process terminates with an error.
The specified server exists in the source HiRDB but not in the system manager unit.		The command starts the extraction node master process, and then terminates itself normally. The corresponding extraction node master process starts, and then terminates with an input/output error on the extraction server status file.	

hdestate (collect source Datareplicator status information)

Function

The `hdestate` command outputs the current status of the source Datareplicator to the standard output. The command outputs the following information:

- **Common information**
Common information such as the source Datareplicator identifier, total number of nodes, and number of connected nodes.
- **Node information**
Information about a node's status, such as the node host name and the status of the extraction node master process.
- **Extraction information queue file information**
Information about the utilization status of the extraction information queue files.
- **Extraction processing information**
Information about the status of extraction processing, such as the status of the extraction process and the write position in the extraction information queue file.
- **Transmission processing information**
Information about the status of transmission processing, such as the target identifier and the status of the transmission process.

Format

```
hdestate [ -b HiRDB-server-name ][ -s target-identifier ]
```

Options

`-b HiRDB-server-name` ~ <symbolic name of 8 characters>

Specify the HiRDB server name of the extraction environment that is to be processed by the `hdestate` command. The command collects status information for the source Datareplicator that corresponds to the specified server name. Note that you must also specify this option when the source HiRDB is a single server.

`-s target-identifier`

Specify a target identifier in order to collect status information about the source Datareplicator corresponding to the specified target. Specifying `**` for the target identifier will result in an invalid-argument error.

To collect status information about a specified target identifier at a specified server, specify both the `-b` and the `-s` options. These options can be specified in either order.

If you omit all options, the `hdestate` command collects status information for all target identifiers at all servers.

Rules

- If the `hdestate` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.
- You can execute the `hdestate` command only while the source HiRDB is active.
- The source Datareplicator collects status information as of the time the `hdestate` command executes and outputs it to the standard output. Therefore, the output information might not match the actual status of the source Datareplicator.
- The source Datareplicator does not output information for a target identifier specified as an absent number.

Output format

```
*****
**          HiRDB Datareplicator extraction status information          **
**                               Mon Mar 29 16:32:05 2004              **
*****
----- Common information -----
hdeid = fa                               ..... 1.
Shmid : 3502                             ..... 2.
PID   : hdemaster = 4579                  ..... 3.
Total node count = 1   Connecting node count = 1 ..... 4.
nodecontrol : server                      ..... 5.

----- Node information -----
host name = host01                       ..... 1.
  current host name = : host0001         ..... 2.
  Communication status : online          ..... 3.
  Shmid : common = 1162  definf = 1163  msginf = 1164 ..... 4.
  Semid : 923                            ..... 5.
  PID   : hdenodemst = 2247              ..... 6.
```

Server information	
Server name = pbes01	1.
Replication-node id : 01	
Semid : 925	
<TRANSMISSION QUEUE FILE INFORMATION>	
wrap count = 0	2.
qfile001 status : using	3.
qfile002 status : using	
qfile003 status : using	
qfile004 status : unused	
qfile005 status : unused	
max used ratio : 2005/03/24 09:21:18 70%	4.
current used ratio : 2005/03/24 09:51:18 10%	5.
<EXTRACTION INFORMATION>	
Status : active	1.
PID : hdecapture = 2248	2.
HiRDB system log overwrite : not overwrite	3.
Queue write position : qfile003, offset = 4096	4.
Extract delay times : 0000:00:05/0000:10:00	5.
<COMMUNICATION INFORMATION>	
Send name = send01	1.
Status : queue read	2.
PID : hdesender = 2249	3.
Service name : serv01	4.
Send host name : shost01	5.
Send system id : aa	6.
Send delay times : 0000:00:10/0000:10:00	7.
Send count : 3	8.
Send data Transmission count :	
transmission transaction count = 50	9.
transaction count2 = 10	10.
undetermined transaction count = 5	11.
ins = 50, upd = 120, del = 30, purge = 2, event = 0	12.
skipped update information count	
ins = 0, upd = 0, del = 0, purge = 0, event = 0	13.
Queue read position : qfile001, offset = 8192	14.
Queue current pos : qfile001, offset = 12288	15.
Transmission inhibition information	
system id = a1, count = 5	16.
system id = a2, count = 0	
system id = b1, count = 2	

Common information

1. hdeid = fa
Displays the source Datareplicator identifier.
2. Shmid: 3502

Displays the shared memory ID. If the source HiRDB is a parallel server, this is the shared memory ID allocated by the system manager.

3. PID: `hdemaster = 4579`

Displays the extraction master process ID.

4. Total node count = 1 and Connecting node count = 1

Total node count displays the total number of server machines managed by the extraction master process. Connecting node count displays the number of server machines that are connected.

5. `nodecontrol: server`

Displays the node master process control method (`nodecontrol` operand value in the extraction system definition). The node master process control method is displayed as follows:

`unit`: Node master process is started for each unit.

`server`: Node master process is started for each server.

Node information

1. `host name = host01`

Displays the host name for a node.

2. `current host name = host0001`

When `nodecontrol` is `server`, displays the current host name as follows, depending on the communication status output in `Communication status`:

- When `Communication status` is `online`

`current host name`: Host name currently recognized as the node master process to be started

- When `Communication status` is `offline`

`Server name`: Server name

3. `Communication status: online`

Displays the status of communication between the extraction master process and extraction node master process. The communication status is one of the following:

`online`: Connected.

`offline`: Disconnected due to an error.

When the status is `offline`, none of the subsequent information is

output.

4. `Shmid: common = 1162 to msginf = 1164`
 Displays the IDs of the shared memories allocated by the extraction node master process. The following are the types of shared memories:
`common`: Shared memory for process-to-process communication
`definf`: Shared memory for storing definition information
`msginf`: Shared memory for storing message information
5. `Semid: 923`
 Displays the semaphore ID for locking message output that was obtained by the extraction node master process.
6. `PID: hdenodemst = 2247`
 Displays the process ID of the extraction node master process.

Extraction information queue file information

1. `Server name = pbes01 to Semid: 925`
`Server name` displays the source HiRDB's server name, and `Replication-node id` displays the data linkage identifier. `Semid` displays the semaphore ID obtained by the extraction node master process.
2. `wrap count = 0`
 Displays the number of times the extraction information queue file was wrapped since initial start of the source Datareplicator.
3. `qufile001 to status: unused`
 Displays the utilization status for each extraction information queue file specified in the `qufile001` to `qufile016` operands in the extraction environment definition. The following are the utilization statuses:
`using`: In use (when update information is stored, the utilization status is always using)
`unused`: Unused
4. `max used ratio: 2005/03/24 09:21:18 70%`
 Displays the date and time at which the extraction information queue file's usage rate peaked (reached the maximum value) and what that usage rate was.
 The following is the display format when the extraction information queue file has just been initialized or when the maximum usage rate has

just been reset by the hdechgstatus command:

```
max used ratio      : ****/**/** **:***:**      0%
```

5. current used ratio: 2005/03/24 09:51:18 10%

Displays the date and time the hdestate command was executed and the extraction information queue file's usage rate at that time.

If all destinations are in the reduced mode, -- is displayed as the usage rate.

Extraction processing information

1. Status: active

Displays one of the following as the status of the extraction process:

active: Active

not active: Inactive

not active (error): Inactive due to an error

If the status is not active or not active (error), extraction processing information 2 below is not displayed.

2. PID: hdecapture = 2248

Displays the extraction process ID.

3. HiRDB system log overwrite: not overwrite

Displays whether the unextracted log was overwritten in the source HiRDB's system log file:

overwrite: Overwritten

not overwrite: Not overwritten

4. Queue write position to offset = 1024

Displays the write position in the extraction information queue file. `qufilexxx` displays one of the `qufile001` to `qufile016` operands specified in the extraction environment definition that corresponds to the extraction information queue file in which the update information is written. `offset` displays the write position of the update information, expressed as the offset from the beginning of the file.

5. Extract delay times: 0000:00:05/0000:10:00

Displays the amount of time that elapsed from the time update data was

committed at the source HiRDB until the time it was written into the extraction information queue file; also displays the value specified in the extraction definition. The following table shows display formats:

Display format	Description
0000:00:00/ <i>defined-value</i>	Nothing has been extracted from an extraction process.
****:**:**/****:**:**	The value 0 was specified in the <code>extract_delay_limit_time</code> operand in the extraction environment definition.
--:----:--/ <i>defined-value</i>	9999:59:59 was exceeded.

Transmission processing information

1. Send name = send01
 Displays the target identifier specified with one of the `sendid01` to `sendid64` operands in the extraction system definition.
2. Status: queue read
 Displays one of the following as the status of the transmission process:
 init: Being initialized
 queue read: Reading the extraction information queue file
 data transmission: Transmitting data
 hold: Transmission in shutdown status
 not active: Inactive
 not active (error): Inactive due to an error
 If the status is hold or not active (error), transmission processing information 3 below is not displayed.
3. PID: hdesender = 2249
 Displays the transmission process ID. However, if `sendmst` is specified in the `sendcontrol` operand in the extraction system definition, 0 is displayed. If an update-SQL output process is running, this item displays the update-SQL output process ID.
4. Service name: serv01
 Displays the transmission server name.
5. Send host name: shost01
 Displays the host name at the destination.

6. `Send system id: aa`
Displays the target Datareplicator identifier at the destination.
7. `Send delay times: 0000:00:10/0000:10:00`
Displays the amount of time that elapsed until reception of update data was completed at the target system from the time it was committed at the source HiRDB; also displays the value specified in the transmission definition. The following table shows display formats:

Display format	Description
<code>0000:00:00 / defined-value</code>	Nothing has been sent by a transmission process.
<code>****: **: ** / ****: **: **</code>	The value 0 was specified in the <code>send_delay_limit_time</code> operand in the transmission environment definition.
<code>--:----:-- / defined-value</code>	9999:59:59 was exceeded.
Not displayed	The update-SQL output facility is used.

8. `send count: 3`
Displays the number of transmissions since the start of transmission processing.
9. `transmission transaction count = 50`
Displays the number of transmission transactions since the start of transmission processing. When `false` is specified in the `send_counter_reset` operand in the extraction system definition, the settings used during the previous session are inherited even when the source Datareplicator is restarted.
10. `transaction count2=10`
Displays the number of transactions created and transmitted by the source Datareplicator separately from those transactions existing when the source database was updated. This count is not included in the number of transmission transactions in `transmission transaction count`.

For example, for a table for which the `WITHOUT ROLLBACK` option is specified, one update operation is treated as one transaction. Therefore, if a table for which the `WITHOUT ROLLBACK` option is specified and a table for which the `WITHOUT ROLLBACK` option is omitted are both updated within the same transaction, the target Datareplicator sends and imports them as separate transactions. In such a case, the displayed number indicates the number of transactions for the table for which the

WITHOUT ROLLBACK option is specified that were created and sent by the source Datareplicator.

11. undetermined transaction count = 5

Displays the number of unresolved transactions since the start of transmission processing.

12. ins = 50 to event = 0

Displays the number of transmitted data items for each type of data manipulation since the start of transmission processing. When `false` is specified in the `send_counter_reset` operand in the extraction system definition, the settings used during the previous session are inherited even when the source Datareplicator is restarted.

The data transmission count for each operation is displayed as follows:

ins: Number of insert data items

upd: Number of update data items

del: Number of delete data items

purge: Number of purge table data items

event: Number of events issued

13. ins = 0 to event = 0

Displays the skipped data count for each type of operation since transmission processing started.

The skipped data count is displayed for each type of operation as follows:

ins: insert data count

upd: update data count

del: delete data count

purge: purge table data count

event: Number of events issued

14. Queue read position to offset = 33792

Displays the read position in the extraction information queue file. `qfilexxx` displays one of the `qfile001` to `qfile016` operands specified in the extraction environment definition that corresponds to the extraction information queue file in which the update information is being read. `offset` displays the read position of the update information, expressed as the offset from the beginning of the file.

In the case of an update-SQL output process, the value of `Queue read position` is not updated.

15. `Queue current pos`

Displays the read location in the extraction information queue file. This information is displayed only when the update-SQL output process is active.

16. `system id = a1 to count = 5`

Displays transmission suppression information for the update information. `system id` displays the system identifier of the original receiver of the update information. `count` displays the number of transmission-suppressed data items.

Notes:

- The values displayed in item No. 2 in the extraction information queue file information and in item Nos. 8 and 11 in the transmission processing information are reset to 0 when they exceed 4294967295.
 - The values displayed in item Nos. 9, 12, and 13 in the transmission processing information are reset to 0 when they exceed 18446744073709551615.
 - Item Nos. 7, 8, 9, 11, and 15 in the transmission processing information are reset each time the source Datareplicator is started.
 - The values displayed in item Nos. 9, 10, 12, and 13 in the transmission processing information are reset at the following times:
 - The source Datareplicator is initialized (including partial initialization) or started[#]
 - An event specified in the `eventcntreset` operand in the transmission environment definition is detected
- [#]: Applicable when `false` is specified in the `send_counter_reset` operand in the extraction system definition
- If update information for `UPDATE` that contains the `SET`, `ADD`, or `DELETE` clause is sent for a repetition column, the `update data count` displayed in item No. 12 in the transmission processing information will be greater than the actual number of updates. If you want to synchronize the `update data count` displayed in item No. 12 and the actual number of updates for repetition columns, update for each clause by using `UPDATE`.

hdestop (terminate the source Datareplicator)

Function

The `hdestop` command terminates the source Datareplicator in the forced termination mode. To terminate the source Datareplicator in the normal termination mode, specify `true` in the `syncterm` operand in the extraction system definition.

Format

```
hdestop [ -t sendterm ]
         [ -e ]
         [ -s [ target-identifier [ { { ,target-identifier } } ... ] ] ]
         [ -w ]
```

Options

`-t sendterm`

Specify this option to cancel processing forcibly in order to terminate transmission processing. This option is applicable when transmission processing only is active and the option to terminate transmission processing (when neither the `-e` nor the `-s` option is specified, or when the `-s` option is specified) had been specified.

If you omit the `-t` option, the source Datareplicator terminates transmission processing after sending all update information that was read from the extraction information queue file during the transmission interval.

`-e`

Specify this option to terminate only extraction of update information from the HiRDB system log file.

`-s [target-identifier [{ { ,target-identifier } } ...]]`

Specify this option to terminate transmission processing only for specified destinations.

target-identifier

To terminate transmission processing for a specified target identifier, specify the target identifier (this must be a target identifier that is specified in the extraction system definition). If you specify multiple target identifiers, do not specify any spaces before or after a delimiter comma. Specifying `**` as the target identifier will result in an invalid-argument error.

If you do not specify any target identifiers, the source Datareplicator terminates transmission processing for all targets.

`-w`

Specify this option to cause `hdestop` command processing not to terminate after it sends a termination request to the source Datareplicator's master process (`hdemaster`) until after `hdemaster` has terminated.

Specify this option when you need to terminate the command after the Datareplicator has terminated completely, such as when a Datareplicator termination shell is used for planned system switchover.

If the `-w` option is omitted, the `hdestop` command terminates itself after sending a termination request to the master process (the master process and the command terminate asynchronously).

If the `-w` option is specified, the `hdestop` command sends a termination request to the master process, waits until the master process terminates, and then terminates itself (when the command terminates, the master process has already terminated).

The following table shows the permitted combinations of `hdestop` command options:

hdestop command option	-t	-e	-s	-w
-t	--	Y	Y	Y
-e	Y	--	Y	--
-s	Y	Y	--	--
-w	Y	--	--	--

Y: Permitted combination.

--: Combination not permitted (if executed with this combination, an error message will be issued).

Rules

- You can have the source Datareplicator terminate automatically when the source HiRDB terminates normally, or you can terminate the source Datareplicator manually by entering the `hdestop` command. The following table shows the relationship between the source Datareplicator's termination method and the termination mode.

Table 7-9: Relationship between the source Datareplicator's termination method and the termination mode

Termination method	Termination mode	Termination processing
Automatic termination when the source HiRDB terminates normally	Normal termination	The source Datareplicator terminates when all the following conditions are satisfied: <ul style="list-style-type: none"> • The source HiRDB's normal termination is detected. • No transactions have occurred at the source HiRDB since detection of the source HiRDB's normal termination. • The extraction process has completed extracting all update information from the system log into the extraction information queue file. • The transmission process has completed sending all update information from the extraction information queue file to the target system. (In the case of transmission delay start, update information in the extraction information queue file is not sent unless the transmission process is started before the extraction process detects the source HiRDB's normal termination.) To have the source Datareplicator terminate when the source HiRDB terminates normally, you must specify <code>true</code> in the <code>syncterm</code> operand in the extraction system definition.
Execution of the <code>hdestop</code> command	Forced termination	When the <code>hdestop</code> command is executed, the source Datareplicator terminates upon completion of the current extraction and transmission processing.

- If you execute the `hdestop` command when `true` is specified in the `syncterm` operand in the extraction system definition, the source Datareplicator terminates in the forced termination mode according to the `hdestop` command.
- You can execute the `hdestop` command only while the source Datareplicator is active. An error results if you execute the `hdestop` command while the source Datareplicator is inactive.
- If the `hdestop` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.

Notes

- If a termination request is issued to an extraction or transmission process that has already been terminated, the termination processing is not executed (because termination processing can be executed only on an active extraction or transmission process). In such a case, the `hdestop` command terminates normally (however, the command outputs an event to that effect to the error information file).
- Once repeated execution of termination processing on extraction or transmission processes results in termination of all extraction and transmission processes, the

`hdestop` (terminate the source Datareplicator)

source Datareplicator terminates.

- If you specify `sendmst` in the `sendcontrol` operand in the extraction system definition to terminate only a specific transmission target, the transmission interval for other transmission targets is cancelled when the `hdestop` command is accepted, and the transmission processing is then started.
- The `hdestop` command outputs the `KFRB04411-E` message and terminates with an error when all of the following apply: `server` is specified in the `nodecontrol` operand in the extraction system definition, the extraction node master process is active, and the extraction master process is inactive. The active extraction node master process does not terminate.

hdestop_n (partially terminate the source Datareplicator)

Function

The `hdestop_n` command partially terminates the source Datareplicator.

The following table provides an overview of this command's operation:

-b option	Process to be terminated			
	Extraction master process	Extraction master process	Extraction process	Transmission process
Omitted	Y#1, #2	--	--	--
Specified	--	Y#1, #3, #4	Y#4	Y#4

#1: If the process is not active, the command terminates with an error.

#2: After the extraction master process has terminated, any active extraction node master process waits for restart of the extraction master process; it waits the amount of time specified in the `connection_retry_waittime` operand. If communication cannot be established with the extraction master process within the specified amount of time, the extraction node master process terminates automatically.

#3: After all extraction node master processes have terminated, any active extraction master process waits for restart of an extraction node master process; it waits the amount of time specified in the `connection_retry_waittime` operand. If it cannot communicate with any extraction node master process within the specified amount of time, it terminates automatically.

#4: The processes to be terminated are the extraction node master process that corresponds to the back-end server specified in the `-b` option and any extraction and transmission processes under it. The command does nothing to the processes in other back-end servers.

Format

```
hdestop_n [ -b server-name [ -t sendterm ] [ -c initial | continue ] ] [ -w ]
```

Options

```
-b server-name [ -t sendterm ] [ -c initial | continue ]
```

Specifies that the extraction node master process at the source Datareplicator is to be terminated. When this option is omitted, the extraction master process is terminated.

-b *server-name*

~ <identifier of 1-8 characters>

Specifies the name of the back-end server that corresponds to the extraction node master process to be terminated.

If the extraction master process or extraction node master process to be terminated is not found, the `hdestop_n` command terminates with an error.

-t `sendterm`

Specifies that transmission processing is to be terminated forcibly. This option is applicable only when a transmission process is active under the extraction node master process that is to be terminated.

When this option is omitted, the process is terminated at the time of completion of the transmission of the update information that was read from the extraction information queue file during the transmission interval.

-c initial | `continue`

When only the extraction or transmission process currently active as the result of a command request is to be started when the next source Datareplicator is started, this option is used to terminate the process.

For details about the operation of extraction and transmission processes that depend on the `-c` option, see the description of the `-c` option in the `hdestart_n` command (partially start the source Datareplicator).

-w

Specifies that the `hdestop_n` command is not to terminate until the target process has terminated.

You specify this operand in order to terminate the `hdestop_n` command after Datareplicator has been terminated completely, such as when a Datareplicator termination shell is used for planned system switchover. When this option is specified, the target process must have terminated in order for the `hdestop_n` command to terminate.

When this option is omitted, the `hdestop_n` command terminates immediately after a termination request has been issued to the target process (the target process and the `hdestop_n` command terminate asynchronously).

Rules

- If the `hdestop_n` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.
- To execute the `hdestop_n` command, you must specify `server` in the `nodecontrol` operand in the extraction system definition (however, normal

execution might be possible; for details, see the rules for the *hdestart_n* command (partially start the source Datareplicator).

- The following table lists and describes the *-b* option value and the operation after execution of the *hdestop_n* command.

Table 7-10: *-b* option value and operation after execution of the *hdestop_n* command

Machine where command is executed		-b option value	Operation after execution of the hdestop_n command
System manager unit	Current	Omitted	The command terminates only the extraction master process, and then terminates itself normally.
		The specified server does not exist in the source HiRDB.	The command terminates with an error because the error information file cannot be found.
		The specified server exists in the source HiRDB but not in the system manager unit.	
		The specified server exists in the source HiRDB and in the system manager unit.	The command terminates only the corresponding extraction node master process, and then terminates itself normally.
	Standby	Omitted	The command terminates with an error because the extraction master process or extraction node master process to be terminated does not exist.
		The specified server does not exist in the source HiRDB.	The command terminates with an error because the error information file cannot be found.
		The specified server exists in the source HiRDB but not in the system manager unit.	
		The specified server exists in the source HiRDB and in the system manager unit.	The command terminates with an error because the error information file cannot be found or the process to be terminated does not exist.
Non-system manager unit	Current	Omitted	The command terminates with an error because the extraction system definition file cannot be found.
		The specified server does not exist in the source HiRDB.	The command terminates with an error because the error information file cannot be found.
		The specified server exists in the source HiRDB but not in the system manager unit.	

hdestop_n (partially terminate the source Datareplicator)

Machine where command is executed		-b option value	Operation after execution of the hdestop_n command
		The specified server exists in the source HiRDB and in the system manager unit.	The command terminates only the corresponding extraction node master process, and then terminates itself normally.
	Standby	Omitted	The command terminates with an error because the extraction system definition file cannot be found.
		The specified server does not exist in the source HiRDB.	The command terminates with an error because the error information file cannot be found.
		The specified server exists in the source HiRDB but not in the system manager unit.	
		The specified server exists in the source HiRDB and in the system manager unit.	The command terminates with an error because the error information file cannot be found or the extraction node master process does not exist.

hdsagtopt (manipulate Datareplicator agent settings)

Function

The `hdsagtopt` command changes settings of the Datareplicator agent on the machine subject to monitoring. When you execute the `hdsagtopt` command, a message is output to the system log and event log indicating that the Datareplicator agent options have been changed. Note that if a percent sign (%) is specified, it will not be output to the system log or event log.

Only the superuser can execute the `hdsagtopt` command.

Format**Changing the monitoring interval**

```
hdsagtopt -t interval
```

Changing the symbol display method

```
hdsagtopt -d tree | notree
```

Changing the symbol mode

```
hdsagtopt -m { e|j|u }
```

Specifying a custom symbol name

```
hdsagtopt -ss source-symbol-name | -ts target-symbol-name
```

Options**Changing the monitoring interval**

```
hdsagtopt -t interval
```

Changes the status monitoring interval. The specifiable value range is 1 to 3600 (seconds). The initial value is 300 (5 minutes). When this command is executed, Datareplicator uses the existing monitoring interval once, and then begins using the new interval.

If the monitoring interval is too small, a lock error might occur during execution of a Datareplicator command.

Changing the symbol display method

```
hdsagtopt -d tree|notree
```

Changes the symbol display method. Specify one of the following arguments:

```
tree
```

Display as a hierarchical structure. This setting is applicable to JP1/Cm2/

Network Node Manager's SNMP agent version 05-00 or later.

notree

Displays as a non-hierarchical structure. This setting is applicable to JP1/Cm2/Network Node Manager's SNMP agent versions earlier than 05-00.

Changing the symbol mode

`hdsagtopt -m { e | j | u }`

Changes the first characters of symbol names. When this command is executed, Datareplicator deletes the existing symbols and creates new symbols. The initial setting is `e`. When you specify `u`, you are then able to specify your own, custom symbol names.

Option	Source symbol name	Target symbol name
<code>e</code>	Datareplicator_source	Datareplicator_target
<code>u</code>	hde (initial value)	hds (initial value)

Specifying a custom symbol name

`hdsagtopt -ss source-symbol-name | -ts target-symbol-name`

Specify a user-specific symbol name. Specify this option in the format `hdsagent -ss source-symbol-name` or `hdsagent -ts target-symbol-name`. A custom symbol name can consist of up to 50 double-byte characters or 100 single-byte characters.

The *source-symbol-name* and the *target-symbol-name* cannot be the same; if they are, the KFRB06105-E error results.

When you execute this command while the symbol mode is `u`, Datareplicator deletes the existing symbols and creates new symbols.

hdsagtstart (start the Datareplicator agent)

Function

The `hdsagtstart` command starts the Datareplicator agent on the machine subject to monitoring. When you execute the `hdsagtstart` command, a message is output to the system log or event log indicating that the Datareplicator agent has been started.

Only the superuser can execute the `hdsagtstart` command.

Format

`hdsagtstart`

Rules

- Execute the `hdsagtstart` command on the Datareplicator that is monitored by JP1/Cm2/Network Node Manager's supervisor machine (manager).
- An error results if you execute the `hdsagtstart` command on a Datareplicator whose agent has already been started.
- To have the `hdsagtstart` command execute automatically when Datareplicator starts, you must do the following in advance:

UNIX Datareplicator:

1. Specify the `$OV_BIN` environment variable for JP1/Cm2/Extensible SNMP Agent.
2. Register `hdsagtstart_sh` in `/etc/localrc`.
3. Register `/opt/hirdbds/lib/hdsagtstart_sh` in `/etc/localrc`.

Windows Datareplicator:

In Windows's Service dialog box, set the HiRDB Datareplicator (Agent) service to **Automatic**.

hdsagtstatus (display the Datareplicator agent status)

Function

The `hdsagtstatus` command displays the status of the Datareplicator agent on the machine subject to monitoring. General users can execute the `hdsagtstatus` command.

Format

`hdsagtstatus`

Output format

```
*****
**          HiRDB Datareplicator agent status information          **
**          Fri Mar 29 16:32:07 2002                             **
*****

agent status : active ..... 1.
interval timer(S) : s ..... 2.
symbol display : tree ..... 3.
symbol mode : e ..... 4.
source site symbol id : Datareplicator_source ..... 5.
target site symbol id : Datareplicator_target ..... 6.
```

Legend:

1. agent status

Displays the status of the Datareplicator agent:

starting: Starting

active: Active (parentheses enclose the process ID)

stopping: Stopping

stop: Stopped

retry: Retrying due to an error

error stop: Detailed information (if there is no detailed information, see the system log or event log)

- Cm2 systemtrap error (NETM*Cm2 system error)
- Insufficient memory (insufficient memory)
- File I/O error (file I/O error)

2. `interval timer`
Displays the monitoring interval (seconds).
3. `symbol display`
Displays the symbol display method:
`tree`: Hierarchical display
`notree`: Non-hierarchical display
4. `symbol mode`
Displays the symbol mode.
5. `source site symbol id`
Displays the symbol name of the source Datareplicator (the symbol name set by the user when the symbol mode is `u`).
6. `target site symbol id`
Displays the target Datareplicator's symbol name (set by the user when the symbol mode is `u`).

hdsagtstop (terminate the Datareplicator agent)

Function

The `hdsagtstop` command terminates the Datareplicator agent on the machine subject to monitoring. When you execute the `hdsagtstop` command, a message is output to the system log or event log indicating that the Datareplicator agent has been terminated.

Only the superuser can execute the `hdsagtstop` command.

Format

`hdsagtstop`

Rules

An error results if you execute the `hdsagtstop` command on a Datareplicator whose agent has already been terminated.

hdscnvedt (edit a mapping table for converting character codes)

Function

The `hdscnvedt` command edits a mapping table for converting character codes.

You use this command to change the Gaiji character conversion method. The `hdscnvedt` command provides the following functions:

- Updating the mapping table for converting character codes

The `hdscnvedt` command changes the conversion method for Gaiji characters that are defined in the mapping table for converting character codes.

- Referencing the mapping table for converting character codes

The `hdscnvedt` command outputs the conversion method for the Gaiji characters that are defined in the mapping table for converting character codes. You can use this output for updating the mapping table for converting character codes.

- Migrating a Gaiji character mapping file

The `hdscnvedt` command imports into the mapping table for converting character codes a Gaiji character mapping file created with Datareplicator version 05-02 or earlier.

Format

Updating the mapping table for converting character codes

```
hdscnvedt -w
           -f source-character-code-set
           -t target-character-code-set
           -d conversion-definition-filename
```

Referencing the mapping table for converting character codes

```
hdscnvedt -r
           -f source-character-code-set
           -t target-character-code-set
           -o output-filename
           [ -s output-start-code ]
           [ -e output-end-code ]
```

Migrating a Gaiji character mapping file

```
hdscnvedt -c
           -f source-character-code-set
           -t target-character-code-set
           -g Gaiji-character-mapping-filename
```

Options

-w

Specify this option to update the mapping table for converting character codes. This option enables you to change the character code correspondences in the range equivalent to Gaiji characters in the mapping table for converting character codes.

-r

Specify this option to reference the mapping table for converting character codes.

-c

Specify this option to migrate definitions from an existing Gaiji character mapping file into the mapping table for converting character codes. This option imports into the mapping table for converting character codes the contents of a Gaiji character mapping file created with Datareplicator version 05-02 or earlier.

-f *source-character-code-set*

Specify the mnemonic for the source character set:

eck78: EBCDIC/KEIS78

eck83: EBCDIC/KEIS83

ekk78: EBCDIK/KEIS78

ekk83: EBCDIK/KEIS83

sjis: JIS8/Shift JIS

euc: EUC

ucs2: UCS2[#]

[#]: If the character code set is UTF-8, specify the UCS2 code set (UTF-8 is managed as the UCS2 code set in the Datareplicator mapping table; actual encoding from UCS2 to UTF-8 is executed during code conversion).

-t *target-character-code-set*

Specify the mnemonic for the target character set:

ekk83: EBCDIK/KEIS83

sjis: JIS8/Shift JIS

euc: EUC

ucs2: UCS2[#]

[#]: If the character code set is UTF-8, specify the UCS2 code set (UTF-8 is

managed as the UCS2 code set in the Datareplicator mapping table; actual encoding from UCS2 to UTF-8 is executed during code conversion).

The following table shows the permitted combinations of character code sets that can be specified in the `-f` and `-t` options:

-f option	-t option						
	eck78	eck83	ekk78	ekk83	sjis	euc	ucs2
eck78	N	N	N	N	Y	Y	Y
eck83	N	N	N	N	Y	Y	Y
ekk78	N	N	N	N	Y	Y	Y
ekk83	N	N	N	N	Y	Y	Y
sjis	Y	Y	Y	Y	N	Y	Y
euc	Y	Y	Y	Y	Y	N	Y
ucs2	N	N	N	N	Y	Y	N

Legend:

Y: Permitted

N: Not permitted (if specified, the KFRB04104-E message is displayed).

`-d conversion-definition-filename`

~ <pathname of 1-127 characters>

Specify the name of the conversion definition file containing the update information for the mapping table for converting character codes, as an absolute or relative pathname. If you specify a relative pathname, Datareplicator assumes the current directory. For details about the definitions in the conversion definition file, see *Format of conversion definition file* below.

`-o output-filename`

~ <pathname of 1-127 characters>

Specify the name of the file to which the mapping table for converting character codes referencing results are to be output, as an absolute or relative pathname. If you specify a relative pathname, Datareplicator assumes the current directory.

Datareplicator outputs the results of referencing the mapping table for converting character codes to the output file in ascending order of the source character codes in the same definition format as the conversion definition file. Datareplicator outputs only those codes for which target codes are defined.

-s output-start-code

This option is applicable only when the *-r* option is specified. It specifies as a hexadecimal character string the source character code with which referencing is to begin.

When this option is omitted, Datareplicator starts referencing at the beginning of the mapping table for converting character codes. You must specify an output start code according to the following rules:

- Specify two characters for a one-byte code, four characters for a two-byte code, and six characters for a three-byte code (applicable when the source character set is EUC or EUC-HJ). Specifying any other number of characters will result in an error.
- If you specify 3-byte EUC-HJ or EUC as the source character code set, make sure that the codes begin with 8f. Specifying any other character will result in an error.
- This value must be smaller than the output end code value specified with the *-e* option. An error will result if this value is greater than the output end code value.

-e output-end-code

This option is applicable only when the *-r* option is specified. It specifies as a hexadecimal character string the source character code with which referencing is to end.

When this option is omitted, Datareplicator references through the end of the mapping table for converting character codes. You must specify an output end code according to the following rules:

- Specify two characters for a one-byte code, four characters for a two-byte code, and six characters for a three-byte code (applicable when the source character set is EUC). Specifying any other number of characters will result in an error.
- A three-byte code must begin with 8f. Specifying any other character will result in an error.
- This value must be greater than the output start code value specified with the *-s* option. An error will result if this value is smaller than the output start code value.

-g Gaiji-character-mapping-filename

~ <pathname of 1-127 characters>

Specify the name of the Gaiji character mapping file that is to be migrated, as an absolute or relative pathname. If you specify a relative pathname, Datareplicator

assumes the current directory.

Format of conversion definition file

Specify in your conversion definition file in the following format each source character code in the mapping table for converting character codes that you want to update and its target character code:

conversion-source-character-code , conversion-target-character-code

When you create a conversion definition file, you must observe the following rules:

- Define one entry per line.
- Specify the conversion source and target character codes as hexadecimal character strings (a through f can be either uppercase or lowercase).
- Begin a comment with the hash mark (#). To enter a comment on the same line as a conversion definition, you must specify at least one space or tab between the definition and the comment delimiter hash mark (#).
- When you specify a character code, specify two characters for a one-byte code, four characters for a two-byte code, and six characters for a three-byte code (applicable when the source character set is EUC or EUC-HJ). Specifying any other number of characters will result in an error.
- A three-byte code must begin with 8f. Specifying any other character will result in an error.
- The first space character that follows the specification of the source character codes is ignored.
- Datareplicator ignores any spaces or tabs before or after a character code.

Rules

- The permitted range of Gaiji characters is as follows:
 - KEIS codes: Byte 1 is from 41 to A0; byte 2 is from A1 to FE.
 - Shift JIS codes: Byte 1 is from F0 to FC; byte 2 is from 40 to FC (excluding 7F).
 - EUC: Byte 1 is 8F; byte 2 is from A1 to FE; byte 3 is from A1 to FE.
 - UCS2: Byte 1 is from E0 to F8; byte 2 is from 00 to FF.
- The superuser must execute this command.
- If you specify the same option more than once, an option specification error results.
- If you specify the same character set for both the conversion source and the target, an error results and the mapping table for converting character codes is not

updated.

- If an error occurs during the command's execution, the mapping table for converting character codes is not updated.
- When the mapping table for converting character codes is updated, Datareplicator stores the new mapping table as a user-created mapping table for converting character codes in *installation-directory/lib/usermap*. Do not delete the mapping table for converting character codes. If there is no mapping table for converting character codes under *installation-directory/lib/usermap*, Datareplicator will convert Gaiji character codes according to its character code conversion rules. For details about Datareplicator's character code conversion rules, see *4.3.5 Designing for character code sets*.
- If you update the mapping table for converting character codes after starting data linkage and the character code conversion results for a mapping key do not match because of the updating, import processing will not execute correctly. Exercise caution in updating the mapping table for converting character codes after data linkage has started.
- Table 7-11 shows the combinations of source and target character sets that you can specify for migrating a Gaiji character mapping file (-c option); specifying any other combination will result in an error:

Table 7-11: Combinations of source and target character sets that you can specify for migrating a Gaiji character mapping file (-c option)

Conversion source character set	Conversion target character set
EBCDIC/KEIS78 EBCDIC/KEIS83 EBCDIK/KEIS78 EBCDIK/KEIS83	JIS8/Shift JIS
JIS8/Shift JIS	EBCDIC/KEIS78 EBCDIC/KEIS83 EBCDIK/KEIS78 EBCDIK/KEIS83
	EUC
EUC	JIS8/Shift JIS

hdschgstatus (change the status of the target Datareplicator)

Function

The `hdschgstatus` command changes the maximum usage rate of the import information queue file. The information changed by this command is retained beyond the termination or start of the target Datareplicator.

Format

```
hdschgstatus  -d data-linkage-identifier
               -c reset  -k max_ratio
```

Options

`-d data-linkage-identifier`

Specify the data linkage identifier that is to be changed.

`-c reset`

Specifies that the information specified in `-k` is to be reset.

`k max_ratio`

Specifies that the maximum usage rate for the import information queue file is to be changed.

Rules

If the `hdschgstatus` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.

Notes

You can execute the `hdschgstatus` command only while the target Datareplicator is active. An error results if you execute it while the target Datareplicator is inactive.

hdsfcopy (copy the current target file)

Function

If a file duplexing error occurs in the target system, disabling one of the physical files (which is placed on `HOLD` status), the `hdsfcopy` command copies from the normal file into the erroneous file.

This processing restores the erroneous file to usable status (`ACTIVE` status).

Format

```
hdsfcopy -f source-physical-file-name -t target-physical-file-name
```

Options

`-f` *source-physical-file-name*

Specify the absolute path name of the source (normal) physical file.

`-t` *target-physical-file-name*

Specify the absolute path name of the target (erroneous) physical file.

Rules

- Execute the command for copying the current file at the node that contains the file that is to be restored.
- The current file copy command is executable when Datareplicator is inactive.
- A command error occurs if the source or target physical file name is not defined correctly in the duplexing definition.
- A command error occurs if the status of the source physical file is not `Active`[#].
- A command error occurs if the status of the target physical file is not `Hold`[#].

#

To check the status of the physical files, use the duplexed target files status display command (`hdsfstate`). For details, see the `hdsfstate` command.

- If a file to be copied is a large file, the command might take a long time to execute. For details, see *6.11.2 Estimating the command execution time when large files are used*.

hdsfmkfs (initialize a Datareplicator file system area)

Function

The `hdsfmkfs` command initializes a Datareplicator file system area.

UNIX Datareplicator supports Datareplicator file system areas. Windows Datareplicator does not support this command.

When you attempt to re-initialize a character special file used with Datareplicator, a message is displayed asking that you confirm that you want to re-initialize the file. If you enter `Y` to this message, the Datareplicator file system area is re-initialized. If you enter any other response, the command is cancelled. If you execute the command specifying a character special file that has not been initialized, the command initializes the file without displaying the confirmation message.

Format

```
hdsfmkfs  -l maximum-files-count -f character-special-filename
           [ -n Datareplicator-file-system-area-size ]#
           [ -q resp|noresp ]
```

#: This option is optional for HP-UX Datareplicator only.

Options

`-l maximum-files-count`

~ <unsigned integer> ((1-255))

Specify the maximum number of files that can be stored in the Datareplicator file system area.

`-f character-special-filename`

~ <pathname>

Specify by its pathname the character special file that is to be initialized. This pathname must be for a Datareplicator file system area. The file size permitted for a Datareplicator file system area depends on the size of the character special file.

`-n Datareplicator-file-system-area-size`

~ <unsigned integer> ((1-1048575)) (MB)

Specify the size of the Datareplicator file system area in MB. If this value is greater than the size of the character special file specified with the `-f` option, the `KFRB04001-E` error results (`errno=28: insufficient disk space`) and the processing is cancelled.

You can omit this option only for HP-UX Datareplicator, in which case the total

size of the character special file specified with the `-f` option is assumed.

If you specify 2048 (2 GB) or a greater value in this option, the file system used to create the Datareplicator file system area must support large files. For details about the handling of large files, see *6.11 Handling of large files*.

`-q resp | noresp`

Specify the `hdsfmkfs` command's action.

`resp`

Output a confirmation message.

`noresp`

Do not output a confirmation message (the command assumes that the file system area is to be initialized without issuing a confirmation message).

Rules

- The following table shows the relationship between the status of the Datareplicator file system area and the `hdsfmkfs` command's options:

Status of the Datareplicator file system area	hdsfmkfs command option		
	None	-q resp	-q noresp
Uninitialized status	Initializes	Initializes	Initializes
Initialized status	Initializes	Initializes	Initializes
Being used (inactive)	Confirms	Confirms	Initializes
Being used (active)	Stops	Stops	Stops

Initializes: Continues initialization processing without displaying the confirmation message.

Confirms: Displays the `KFRB04809-Q` confirmation message and continues initialization processing if `Y` is entered as the response.

Stops: Displays the `KFRB04807-E` message and cancels initialization processing.

- If the `hdsfmkfs` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.
- If any of the following is applicable to the Datareplicator file system area (according to the message that is output), the `hdsfmkfs` command's processing is cancelled.

The area has not been initialized

- The specified character special file has not been allocated.
- The user executing Datareplicator does not have the `read/write` privilege for the character special file.
- The specified file is not a character special file.

The area is being used

- The `hdsfmkfs` command was executed while Datareplicator was executing.

A file access error occurred

- An error occurred on the disk that contains Datareplicator file system area.

hdsfstate (display the status of duplexed target files)

Function

The `hdsfstate` command displays the status of duplexed files (physical files) in the target system.

Format

`hdsfstate [-f logical-file-name]`

Options

`-f logical-file-name`

Specify the absolute path name of the logical file whose status is to be displayed.

If this operand is omitted, the command displays the status of all duplex files at the server machine where this command is executed.

Output format

The `hdsfstate` command displays the status of the extraction information queue files in the following format:

```

*****
**          HiRDB Datareplicator file status information          **
**          Sun Aug 25 12:58:52 2002                            **
*****
Logical file name = /users/repli/quefile001_bes1                ....1.

sys status  type  Physical file name                            ....2.
a  a-----  rqu  /hd01/hds/qufile001_bes1_1
b  a-----  rqu  /hd01/hds/qufile001_bes1_2
-----
Logical file name = /users/repli/quefile002_bes1

sys status  type  Physical file name
a  a-----  rqu  /hd01/hds/qufile002_bes1_1
b  h-----  rqu  /hd01/hds/qufile002_bes1_2

Number of Logical file Information = 2                        ....3.

```

1. Indicates the logical file name.
2. Indicates status information for the physical file.

`sys`: Physical file system

For the primary system, `a` is displayed; for the secondary system, `b` is displayed:

Output	Description
a	Primary file
b	Secondary file

status: Physical file status. The character in column 1 represents the status:

Output	Description
a	Current file status (ACTIVE)
h	Error status (HOLD)
*	Status cannot be obtained.

type: Type code for the physical file:

Output	Description
rmt	Import master status file
rst	Import status file
rqu	Import information queue file
---	Other file
***	File type cannot be obtained

Physical file name: Name of the physical file

- Indicates the number of logical file information items that have been displayed.

hdsfstatfs (display the status of a Datareplicator file system area)

Function

The `hdsfstatfs` command displays the status of a specified Datareplicator file system area.

UNIX Datareplicator supports Datareplicator file system areas. Windows Datareplicator does not support this command.

You can execute this command regardless of whether Datareplicator is active. If you execute the `hdsfstatfs` command while the Datareplicator file system area is being initialized, it displays the status of the area at the time the command was entered.

Format

`hdsfstatfs -f Datareplicator-file-system-area-name`

Option

`-f Datareplicator-file-system-area-name`

~ <pathname>

Specify the name of the Datareplicator file system area whose status is to be displayed.

Rules

- If the `hdsfstatfs` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.
- If any of the following is applicable to the Datareplicator file system area (according to the message that is output), the `hdsfstatfs` command's processing is cancelled:

The area has not been initialized

The Datareplicator file system area has not been initialized.

The area was not found

The Datareplicator file system area specified in the `hdsfstatfs` command was not found.

A file access error occurred

An error occurred on the disk that contains the Datareplicator file system area.

Output format

The command's execution results are output to the standard output. The following

is the output format:

```

*****
**          HiRDB Datareplicator file system area information          **
**          Fri Mar 29 16:32:05 2002                                **
*****
file system area      /HiRDBDS/sysfile/sysarea05      ..... 1.
status                used                ..... 2.
initialize time       Mon Oct 22 19:40:01 2001 ..... 3.
Datareplicator directory /HiRDBDS/system01/hde ..... 4.
Datareplicator type   hde                ..... 5.
area capacity         204800[KB]          ..... 6.
remain area capacity  44691[KB]          ..... 7.
sector size           1024[B]            ..... 8.
available file count   20                ..... 9.
used file count       18                ..... 10.

[ 1] file name = /HiRDBDS/system01/hde/sts_bes2 ..... 11.
    [ file kind = DS_STATS   file size = 77[KB] ..... 12.
[ 2] file name = /HiRDBDS/system01/hde/sts_bes2.
    file kind = DS_HDEFL    file size = 9[KB].
[ 3] file name = /HiRDBDS/system01/hde/que2/qufile001_bes2.
    file kind = DS_QUEUE    file size = 10001[KB]

      :
[ 17] file name = /HiRDBDS/system01/hde/que2/qufile015_bes2
      file kind = DS_QUEUE   file size = 10001[KB]
[ 18] file name = /HiRDBDS/system01/hde/que2/qufile016_bes2.
      file kind = DS_QUEUE   file size = 10001[KB]

```

Explanation

1. file system area: Name of the Datareplicator file system area
2. status: Status of the Datareplicator file system area:
 - init: Initialized by the hdsfmkfs command
 - used: Being used by Datareplicator
 - error: Error occurred during initialization by the hdsfmkfs command
3. initialize time: Time the Datareplicator file system area was initialized by the hdsfmkfs command
4. Datareplicator directory: Name of the Datareplicator directory used
5. Datareplicator type: Type of Datareplicator:
 - hde: Source Datareplicator
 - hds: Target Datareplicator

hdsfstats (display the status of a Datareplicator file system area)

6. `area capacity`: Total size of the Datareplicator file system area (KB)
7. `remain area capacity`: Amount of unused space in the Datareplicator file system area (KB)
8. `sector size`: Sector size of the Datareplicator file system area (KB)
9. `available file count`: Maximum number of files specified in the `hdsfmkfs` command
10. `used file count`: Number of existing files
11. `[nn] file name = aa...aa`:
`nn`: Sequence number of a system file in the Datareplicator file system area
`aa...aa`: Name of the system file
12. `file kind = bb...bb file size = cc...cc` [KB]
`bb...bb`: Type of system file:
 - `DS_STATUS`: Status file
 - `DS_HDEFL`: HiRDB communication file
 - `DS_QUEUE`: Queue file`cc...cc`: Size of the system file (KB)

hdspathlist (specify a directory to be monitored)

Function

The `hdspathlist` command specifies the Datareplicator directory on a machine that is to be subject to monitoring when JP1/Cm2 is used for operations management. The supervisor machine monitors the Datareplicators that correspond to the directories specified in the `hdspathlist` commands.

Only the superuser can execute the `hdspathlist` command, except that general users can execute the `hdspathlist -l` command.

There is no need to execute the `hdspathlist` command at the machine subject to monitoring for a Datareplicator that does not use operations management or for Windows Datareplicator.

Format**Initializing the directories subject to monitoring**

```
hdspathlist -i
```

Checking the directories subject to monitoring

```
hdspathlist -l [ -m hde|hds ]
```

Adding or deleting a directory subject to monitoring

```
hdspathlist { -a|-d } -m { hde|hds } -p directory-name
```

Options

`-i`

Initializes the list of directories.

`-l`

Checks the list of Datareplicator directories subject to monitoring. General users can execute the `hdspathlist -l` command.

`-a`

Adds a Datareplicator directory that is to be monitored.

`-d`

Deletes a Datareplicator directory subject to monitoring from the list of directories.

`-m`

Specify the type of Datareplicator to be checked, added, or deleted:

hdspathlist (specify a directory to be monitored)

hde: Source Datareplicator

hds: Target Datareplicator

-p *directory-name*

~ <pathname of up to 255 single-byte characters>

Specify the complete pathname of the Datareplicator directory to be added or deleted.

Output format

```
*****
**      HiRDB Datareplicator extract/reflect directory list      **
*****

-- Reflect Datareplicator directory list --
  /opt/hirdb/TEST/hds0100/recv1
  /opt/hirdb/TEST/hds0100/recv2
  /opt/hirdb/TEST/hds0100/recv3
  /opt/hirdb/TEST/hds0100/recv4
  /opt/hirdb/TEST/hds0100/recv5
}
```

hdsrefinfmt (check update information)

Function

The `hdsrefinfmt` command extracts each update information item from the import information status file and outputs it to the analysis results output file. You use this command for the following purposes:

- To determine whether the import information queue files contain the update information identifiers that are included in the import suppression list file.
- To check the update information that has or has not been imported.

Format

```
hdsrefinfmt -f import-status-file-name
                {-l 9 | -m trn [-t imported-transactions-count ,
                                unimported-transactions-count]}
                [-g import-group-name | -c] -p analysis-results-output-file-name
                [-k maximum-size-of-output-file]
```

Options

Option for specifying the extraction target

`-f import-status-file-name`

Specify the absolute path name of the import status file from which information is to be extracted.

Options for specifying the update information to be output

`-l 9`

Outputs information in the format shown in *Output format 1*.

`-m trn`

Outputs information in the format shown in *Output format 2*.

`-t imported-transactions-count , unimported-transactions-count`

`~<alphanumeric characters> ((0-999999999, all)) <<1,1>>`

If you specify the `-m trn` option, specify the update information to be output as the number of source transactions. Specify the numbers of imported transactions and unimported transactions separated by a comma (.). To output all transactions, specify `all`.

Options for specifying the base point for output of update information

`-g import-group-name`

Sets the beginning of the specified import group as the base point for output of the update information. Specify the absolute path name of the import group.

-c

Sets the beginning of the import process as the base point for output of the update information.

If you omit these options, the command assumes the current import process execution location as the base point.

Options related to output of analysis results

-p *analysis-results-output-file-name*

Specify the absolute path name of the file to which the results are to be output.

-k *maximum-size-of-output-file*

~ <unsigned integer> ((0-1048576)) <<1024>> (KB)

Specify the maximum size of the analysis results output file. If you specify 0, there is no limit to the size of the analysis results file.

Note that the actual size of the analysis results output file might become slightly larger than the specified value (by about 1 KB).

Output format 1

The output format of the `hdsrefinfmt` command is shown below (applicable when the `-l 9` option is specified).

The update information identifier is displayed in the analysis results output file as `Extract-id`, which is one of the import information queue file analysis results items, and the extraction transaction information is displayed as `Additional Transaction information`.

Note that the items other than `Extract-id` and `Additional Transaction information` constitute maintenance information and their contents are not guaranteed.

Output format 2

The output format of the `hdsrefinfmt` command is shown below (applicable when the `-m trn` option is specified).

The update information identifier is displayed in the analysis results output file as `Extract-id`, which is one of the import information queue file analysis results items, and the extraction transaction information is displayed as `Additional Transaction information`.

Note that the items other than `Extract-id` and `Additional Transaction information` constitute maintenance information and their contents are not guaranteed.

```

**** HiRDB Datareplicator Reflection Infomation <Wed Mar 9 20:49:28 2005> ****
*****
**** Reflected Information ****
*****
+-----+
| Tran Data(-2) |
+-----+
Transaction Branch information=(4e4b4442756e30310000004f, 1, 00 )
Additional Transaction information=(Tx,TranInfo, 423a43590000000000000079)
Updating Data(1) ExtractDate(2005/03/09 20:48:07)
Extract Id = 422E5A8A0000000000011EF:1
Table(131193, hitachi.T1) Kind(PURGE) DataType(Row)
+-----+
| Tran Data(-1) |
+-----+
Transaction Branch information=(4e4b4442756e303100000053, 1, 00 )
Additional Transaction information=(Tx,TranInfo, 423a43590000000000000008a)
Updating Data(1) ExtractDate(2005/03/09 20:48:08)
Extract Id = 422E5A8A000000000001200:1
Table(131193, hitachi.T1) Kind(INSERT) DataType(Col)
Column(1, C1) Data-Info(Key, Norm)
After Data : 00000000 *.... *
Column(2, C2) Data-Info(Norm)
After Data : 30202020 20202020 20202020 20202020 *0 *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 * *
**** Summary Information *****
Transaction Information
TRNcount      INScount      UPDcount      DELcount      PRGcount      EVTcount
          2              1              0              0              1              0
*****
**** Un-reflect Information ****
*****
+-----+
| Tran Data(1) |
+-----+
Transaction Branch information=(4e4b4442756e303100000054, 2, 00 21 )
Additional Transaction information=(Tx,TranInfo, 423a43590000000000000008d)
Updating Data(1) ExtractDate(2005/03/09 20:48:09)
Extract Id = 422E5A8A000000000001204:1
Table(131193, hitachi.T1) Kind(UPDATE) DataType(Col)
Column(1, C1) Data-Info(Key, Norm)
Before Data : 00000000 *.... *
Column(1, C1) Data-Info(Key, Norm)
After Data : 00000064 *...d *
Column(2, C2) Data-Info(Norm)
After Data : 31303020 20202020 20202020 20202020 *100 *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 * *

```

```

+-----+
| Tran Data(2          ) |
+-----+
Transaction Branch information={4e4b4442756e30310000005a, 2, 00 21 }
Additional Transaction information={Tx,TranInfo, 423a43590000000000000008e)
Updating Data(1) ExtractDate(2005/03/09 20:48:10)
Extract Id = 422E5A8A0000000000001212:1
Table(131193, hitachi.T1) Kind(INSERT) DataType(Col)
Column(1, C1) Data-Info(Key,Norm)
After Data : 00000001 *.... *
Column(2, C2) Data-Info(Norm)
After Data : 31202020 20202020 20202020 20202020 *1 *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 * *
Updating Data(2) ExtractDate(2005/03/09 20:48:11)
Extract Id = 422E5A8A0000000000001212:2
Table(131193, hitachi.T1) Kind(INSERT) DataType(Col)
Column(1, C1) Data-Info(Key,Norm)
After Data : 00000002 *.... *
Column(2, C2) Data-Info(Norm)
After Data : 32202020 20202020 20202020 20202020 *2 *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 20202020 20202020 20202020 * *
                20202020 * *
Event(201)
+-----+
| Tran Data(3          ) |
+-----+
Transaction Branch information={4e4b4442756e30310000005b, 2, 00 21 }
Additional Transaction information={Tx,TranInfo, 423a435900000000000000090)
Updating Data(1) ExtractDate(2005/03/09 20:48:11)
Extract Id = 422E5A8A0000000000001216:1
Table(131193, hitachi.T1) Kind(DELETE) DataType(Col)
Column(1, C1) Data-Info(Key,Norm)
Before Data : 00000001 *.... *
**** Summary Information ****
Transaction Information
TRNcount      INScount  UPDcount  DELcount  PRGcount  EVTcount
      3             2           1           1           0           1
*****

```

hdsrfctl (control import processing)

Function

The following are the functions of the `hdsrfctl` command:

- Restart of import processing

Import processing restarts using the specified import method. When you execute the `hdsrfctl` command, import processing restarts immediately.
- Termination of import processing

You can terminate import processing while keeping reception processing active. The import processing termination timing depends on the specified options.
- Modification of the import method

To modify the import method while import processing is underway, use the following procedure:

 1. Stop import processing.
 2. Specify either the transaction-based import method or the table-based import method, and then restart import processing.
- Modification of the import processing commit interval

You can change the commit interval for the current import processing. If import processing is stopped, the command restarts import processing with the specified commit interval.

Format 1

```
hdsrfctl -d data-linkage-identifier -m { trn|tbl|spd|immediate }
        [ -c import-processing-commit-interval ]
```

Format 2

```
hdsrfctl -g synchronous-import-group-name
        -m { start|immediate }
```

Options (Format 1)

`-d` *data-linkage-identifier*

Specify the data linkage identifier of the source system that corresponds to the import processing that is to be subject to the command's processing. Specifying `**` for the data linkage identifier will result in an invalid-argument error.

For details about the data linkage identifier when the source database is a HiRDB, see *5.2 Extraction system definition*. For details about the data linkage identifier when the source database is a mainframe database, see the *VOS3 XDM/DS*

manual.

-m { trn|tbl|spd|immediate }

trn

Execute import processing using the transaction-based import method.

tbl

Execute import processing using a table-based import method (table-based partitioning method, key range-based partitioning method, or hash partitioning method).

spd

Terminate import processing when an event issued from the source system is detected during import processing after the command executes.

immediate

Terminate import processing when the import processing that was underway when the command was executed terminates. If update information is being received when the command is executed, import processing terminates when Datareplicator finishes storing the update information in the import information queue file.

-c *import-processing-commit-interval*

~ <unsigned integer> <<value of the *cmtintvl* operand in the import environment definition>> ((1-32767))

Specify the interval at which the import processing subject to the command's processing is to issue commit to the target HiRDB. This value is applied to import processing that starts after execution of the *hdsrfctl* command. If you execute the *hdsstop* command or re-execute the *hdsrfctl* command, Datareplicator will ignore the previous value specified in the *-c* option.

If the *hdsrfctl* command with the *-c* option specified results in an error while restarting import processing or changing the commit interval, the specified commit interval will be ignored.

Options (Format 2)

-g *synchronous-import-group-name*

Specify the synchronous import group to be controlled.

-m { start|immediate }

start

Starts the synchronous import group. If the synchronous import group was disabled during the previous termination, it is started in disabled status.

immediate

Terminates import processing. The termination methods are as follows:

- If a synchronous event is detected during transaction processing, the command commits the transaction, and then terminates the import processing.
- If the end of the import information queue file is detected during transaction processing, the command rolls back the transaction, and then terminates the import processing.
- If no transaction processing is underway, the command terminates the import processing immediately.

Rules

- The `hdsrctl` command controls import processing only for a specific data linkage identifier. To control import processing for multiple data linkage identifiers, you must execute the `hdsrctl` command for each data linkage identifier.
- Execute the `hdsrctl` command specifying a data linkage identifier for each source system.
- Before you can change the import method with the `hdsrctl` command, you must terminate the import processing in the event termination mode using the `hdsrctl -m spd` command.
- The target Datareplicator gives precedence to an `hdsrctl` command request over an event request. For example, if you execute an event requesting modification of the import method as well as an import processing termination request with the `hdsrctl` command, Datareplicator terminates import processing according to the command request.
- When the `hdsrctl` command is executed, the target Datareplicator is placed in event wait status. If you re-execute the `hdsrctl` command in this event wait status, the second `hdsrctl` command executed becomes subject to the event wait status.
- An error results if you execute the `hdsrctl` command while the target Datareplicator is engaged in termination processing due to execution of the `hdsstop` command.
- If import processing terminates abnormally while the target Datareplicator has been placed in event wait status by the `hdsrctl` command, the `hdsrctl` command subject to event wait status is cancelled.
- If you are executing import processing with the delayed start method specifying a delay time (`breaktime` operand), you can invalidate the specified delay time and cancel the delayed start by executing the `hdsrctl -m spd` command, in

which case the import processing will not start at the specified time.

- If some import groups have terminated with an error during import processing in the table-based import mode, you cannot restart only the import groups that have terminated. For details about the handling procedure when some import groups terminate with an error, see *6.7.1 Handling of import processing*.
- If you have specified `spd` or `immediate` in the `-m` option of the `hdsrfctl` command, Datareplicator ignores any commit interval that is specified in the `-c` option.
- If you execute the `hdsstop` or `hdsrfctl` command more than once, the operation depends on the specified options; for details, see *Table 7-16 Datareplicator's action when the hdsstop or hdsrfctl command is executed more than once*.
- The following table shows the `hdsrfctl` command's return values, the termination classifications, and the termination causes:

Return value	Classification	Cause
0	Normal termination	--
1	Abnormal termination	Main causes are as follows: <ul style="list-style-type: none"> • Invalid argument • Invalid command execution environment (such as an invalid environment variable or an invalid definition file) • System call error • Source Datareplicator is inactive
11	Impossible to accept command	Import processing is already executing.
12		There was no response to the import processing start request.

hdssamqin (extract update information from a SAM file)

Function

When the source database uses SAM files (PDM2 E2 or RDB1 R2), the `hdssamqin` command accumulates the update information transferred in a SAM file into the import information queue file. Datareplicator imports the accumulated update information to HiRDB during import processing.

Format

```
hdssamqin -d data-linkage-identifier -f input-SAM-filename
          -n update-information-definition-filename
          [ -l update-information-editing-buffer-size ]
          [ -c ][ -o ]
```

Options

`-d data-linkage-identifier`

Specify the data linkage identifier. This data linkage identifier must be unique among all update information input processes and reception processes that are executed concurrently. Specifying `**` for the data linkage identifier will result in an invalid-argument error.

`-f input-SAM-filename`

~ <*absolute-pathname*>

Specify the input SAM file. This is the SAM file transferred from the source database in binary mode. If you specify only the filename, Datareplicator assumes that the file is located in the current directory where the command was executed.

`-n update-information-definition-filename`

~ <*absolute-pathname*>

Specify the name of the file that contains the update information definition for data extraction. If you specify only the filename, Datareplicator assumes that the file is located in the current directory where the command was executed.

The `-n` option is applicable only to PDM2 E2; you cannot specify the `-n` option for RDB1 E2 because there is no need to create an update information definition file.

`-l update-information-editing-buffer-size`

~ <unsigned integer> ((1-2097151)) <<300>> (KB)

Specify the size of the buffer used to edit update information extracted from PDM2 E2. The `-l` option is applicable only to a SAM file extracted from PDM2

E2; it is not applicable to a SAM file extracted from RDB1 E2.

-c

Specify this operand to ignore the SAM file information that was processed during the previous execution of the command. Datareplicator will ignore the specified SAM file regardless of whether the previous `hdssamqin` command terminated normally or abnormally.

-o

If an error occurred during the previous command execution, specify this option to resume the previous processing.

Rules

If the `hdssamqin` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.

Notes

If the SAM file input processing terminates abnormally and input file processing is not completed, do not execute the Datareplicator's initial startup command (`hdsstart -i`). If `hdsstart -i` is executed, Datareplicator will not be able to establish data linkage correctly, because information about the SAM file input synchronization point is lost. If you have executed initial startup, re-create the target HiRDB table, and then execute the `hdssamqin` command.

hdsshmclean (delete the target Datareplicator's shared resources)

Function

The `hdsshmclean` command deletes shared resources (processes, shared memory, semaphores) that remain when the target Datareplicator terminates abnormally. This command is only applicable to the UNIX edition. The Datareplicator administrator executes this command.

Format

```
hdsshmclean [ -l clean[ -q{resp|noresp}]] [ -w ]]
```

Options

`-l clean`

Specify this option to delete shared resources. If this option is omitted, the command only displays the remaining shared resources.

`-q{resp|noresp}`

Specify whether the `KFRB04331-Q` message is to be displayed for confirmation before the shared resources are deleted.

`resp`

Outputs the `KFRB04331-Q` message. The command deletes the shared resources only when `y` is entered.

`noresp`

Deletes the shared resources without displaying the `KFRB04331-Q` message.

`-w`

Specify this option to delay termination of this command until deletion of the shared resources has been completed. If this option is omitted, whether deletion of the shared resources has been completed when the command terminates cannot be guaranteed.

Output format

```

<<<*** Common information ***>>> ..... 1.
Shmid : common      = 13467653 ..... 2.
Semid : common      = 8912925 ..... 3.
PID   : hdsmaster   = 7044 ..... 4.
PID   : hdstrcrv    = 7046 ..... 5.
PID   : hdstcpmst   = 7047 ..... 6.
PID   : hdsosimst   = 7048 ..... 7.

<<<*** Sync group information (grpname) ***>>> ..... 8.
Shmid : refsyc      = 6422538 ..... 9.
Shmid : TBI-list    = 6684690 6684691 ..... 10.
Semid : refsyc      = 10453040 ..... 11.
PID   : hdsrefsync  = 22707 ..... 12.

<<<*** Replication information (01) ***>>> ..... 13.
Shmid : defserv     = 13533191 ..... 14.
PID   : hdstcpmst   = 27921 ..... 15.
PID   : hdsdefserv  = 27922 ..... 16.

<<<*** Group information (trngroup) ***>>> ..... 17.
Shmid : lobshm      = 0 ..... 18.
PID   : hdsreflect  = 27958 ..... 19.
PID   : hdssqle     = 27959 ..... 20.

<<<*** Replication information (02) ***>>>
Shmid : defserv     = 14254130
PID   : hdsosimst   = 27931
PID   : hdsdefserv  = 27932

<<<*** Group information (trngroup) ***>>>
Shmid : lobshm      = 0
PID   : hdsreflect  = 27960
PID   : hdssqle     = 27963
    
```

Explanation of the output information

No.	Output information	Remarks
1	Header for the resources managed by the master process	--
2	Shared memory ID for process-to-process communication at the target	--
3	Semaphore ID for locking target processes	--
4	Process ID of import master process	--
5	Process ID of activity trace collection process	--

No.	Output information	Remarks
6	Process ID of communication master process (TCP/IP)	--
7	Process ID of communication master process (OSI)	--
8	Header for the synchronous import group resources	This information is output when <code>syncgroup001</code> is specified in the import system definition.
9	Shared memory ID for the synchronous import group	
10	Shared memory ID for TBI list	This information is output when <code>syncgroup001</code> is specified in the import system definition. If there are multiple shared memories for TBI list, the items are delimited by a space.
11	Semaphore IDs for locking within a synchronous import group	This information is output when <code>syncgroup001</code> is specified in the import system definition.
12	Process ID of synchronization managing process	
13	Resource header for each data linkage identifier	--
14	Shared memory ID for storing import definition	--
15	Process ID of data reception process	Depending on the protocol being used, either <code>hdstcpmst</code> or <code>hdsosimst</code> is output for the process name.
16	Process ID of import definition server process	--
17	Resource header for each import group	Nos. 17 through 20 are output for each import group.
18	Shared memory ID for storing BLOB data	--
19	Process ID of import process	If the target Datareplicator was started with the <code>startmode</code> operand setting changed to <code>spd</code> in the import environment definition, the process ID used during the previous session might be displayed.
20	Process ID of import SQL process or import UOC process	

Legend:

--: Not applicable

Rules

If the target Datareplicator cannot be started because there are shared resources remaining, execute the `hdsstop` command to delete the shared resources. Use this command if the shared resources cannot be deleted by executing the `hdsstop` command.

hdsstart (start the target Datareplicator)

Function

The `hdsstart` command starts the target Datareplicator in accordance with its import system definition. For details about the import system definition, see *5.8 Import system definition*.

Format

```
hdsstart [ -i [ init ] [ -D data-linkage-identifier-number ] [ -f ] [ -q ] [
-r ] ]
          | -c synchronous-import-group-name
```

Options

`-i`

Specify this option to first initialize the target Datareplicator environment, and then start the target Datareplicator (initial start).

An error results if, when the `-f` option is omitted, there is still an import environment subject to initialization, and the import information queue file for that import environment contains update information that has not yet been imported.

When the `-D` option is specified, Datareplicator initializes only the import environment for the specified source, and then starts the target Datareplicator. When the `-D` option is omitted, Datareplicator initializes the following files specified in the import system definition or import environment definition, initializes the entire target Datareplicator environment, and then starts the target Datareplicator (when the `-D` option is specified, Datareplicator initializes only those files that correspond to the specified source's data linkage identifier):

- Import information queue files
- Import status files
- Import master status file
- Import error information files
- Import trace files
- Unimported information files

`init`

Specify this option to create all import information queue files and import status files specified in the import environment definitions based on the sizes specified in the `queuesize` and `statssize` operands, respectively.

When you execute the `hdsstart` command with `init` specified, the initialization processing takes more time than when only the `-i` option is specified. This amount of time depends on the file sizes and the number of files to be created.

If this option is specified when large files are used, the command might take a long time to execute. For details, see *6.11.2 Estimating the command execution time when large files are used*.

When the `-D` option is specified, the Datareplicator creates only those files that correspond to the specified source's data linkage identifier.

`-D` *data-linkage-identifier-number*

Initializes the import environment for the specified source only (partial initialization). Datareplicator executes normal startup on the import environments corresponding to sources that are not specified. If an error occurs during initial start processing, Datareplicator cancels the processing, in which case the target Datareplicator does not start.

Specify the source identifier number, which is the numeric digits in *xxx* of the data linkage identifier specified in the import system definition (`dsidxxx`). The specified data linkage identifier number must have been defined when the most recent initial start was executed. You cannot specify a newly added data linkage identifier number. The partial initialization specified in this operand is applicable to import environment definitions specified with `dsidxxx` and `refenvxxx` operands in the import system definition and the import definitions specified with `reffile` in the import environment definitions.

Datareplicator handles any changes to a definition other than partial initialization in the same manner as a normal start.

Datareplicator handles any change to a definition other than partial initialization in the same manner as a normal start.

Do not change a definition other than the data linkage identifier that is subject to partial initialization. To perform partial initialization on multiple data linkage identifiers, first change the definition of a data linkage identifier, and then perform partial initialization; then, repeat this procedure for each additional data linkage identifier.

`-f`

Executes an initial start forcibly, regardless of whether the import information queue file contains unimported update information. If the import information queue file contains unimported update information, that information will be discarded during the initial start. Therefore, when you specify this option to perform an initial start, check first to determine whether there is any unimported update information that will be needed.

Specify this option if any of the following is true:

- Datareplicator was terminated forcibly by the `hdsstop -t force` command.
- Datareplicator is to be initialized for the first time after the import information queue file and the import status file were created using character special files (there is no need to specify this option for a subsequent initialization).
- The status file specified in the `statsfile` operand in the import environment definition is to be renamed and a file with the new status file name already exists.

-q

Terminates the target Datareplicator after initial start or partial initial start without starting it.

-r

Resets the import processing count when the target Datareplicator starts.

This option is applicable when `false` is specified in the `reflect_counter_reset` operand in the import system definition.

-c *synchronous-import-group-name*

Cancels synchronous import of the specified synchronous import group and starts the target Datareplicator normally. You use this option when an import transaction cannot be committed due to failure of the `hdevent` command to execute, or due to a shortage of target database resources.

Import processing can be executed later for each of the data linkage identifiers constituting the cancelled synchronous import group. Additionally, the operand that was disabled by specification of the synchronous import group is now enabled.

Once a synchronous import group is cancelled, it cannot be reorganized again until Datareplicator is initialized. If the specified synchronous import group name is undefined or has already been cancelled, the target Datareplicator results in an error during the normal start.

Rules

- If the `hdsstart` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.
- The start mode depends on the previous termination mode. The following table shows the relationship between the start mode and the previous termination mode.

Table 7-12: Relationship between the start mode and the previous termination mode

Start mode	Previous termination mode	Action at startup
Initial start	N/A ^{#1}	Starts the target Datareplicator in accordance with the import system definition without inheriting the settings used at the previous session. At an initial start, the target Datareplicator initializes files, as shown in Table 7-13 <i>Handling of files when the -i option is specified</i> .
Partial initial start	N/A ^{#2}	At partial initial start, the target Datareplicator initializes the import environment only for the specified source and starts operation.
Normal start	Normal termination, event termination, or immediate termination	The target Datareplicator starts in accordance with the import system definition without inheriting the settings used at the previous session.
Restart	Abnormal termination or forced termination	The target Datareplicator ignores some of the operand settings ^{#3} in the import environment definition and starts operation with the settings used at the previous session in order to guarantee conformity between the source and target databases.

#1: Executing the `hdsstart` command with the `-i` option specified starts the target Datareplicator in the initial start mode, regardless of the previous termination mode.

#2: Datareplicator executes initial start on the processing for the data linkage identifier that is specified in the `-D` option as being subject to initialization. On the processing for any other data linkage identifier, Datareplicator executes either normal start or restart according to the previous termination mode.

#3: In the restart mode, Datareplicator ignores the settings of the `startmode`, `breaktime`, `breakmode`, `eventspd`, `eventtrn`, `eventtbl`, `cmtintvl`, `trncmtintvl`, and `tblcmtintvl` operands in the import environment definition. For the `cmtintvl`, `trncmtintvl`, and `tblcmtintvl` operands, their settings take effect automatically when conformity is guaranteed between the source and target databases.

- The following table shows the handling of each file when the `-i` option is specified.

Table 7-13: Handling of files when the -i option is specified

File type	No existing file	Type of existing file		Datareplicator file system area ^{#1}	
		OS regular file	Character special file ^{#1}	In initialized status	Being used
Import information queue file ^{#2}	Created in the OS regular file format	Re-created	Does nothing. ^{#3}	File is allocated.	File is reallocated.
Import status file ^{#2}			Header is initialized. ^{#3}	File is allocated.	File is reallocated.
Import master status file			Header is initialized. ^{#3}	N/A	N/A
Unimported information file			N/A	N/A	N/A
Import error information file			N/A	N/A	N/A
Import trace file			N/A	N/A	N/A

#1: UNIX Datareplicator supports character special files and Datareplicator file system areas.

Windows Datareplicator executes the processing described in the *OS regular file* column.

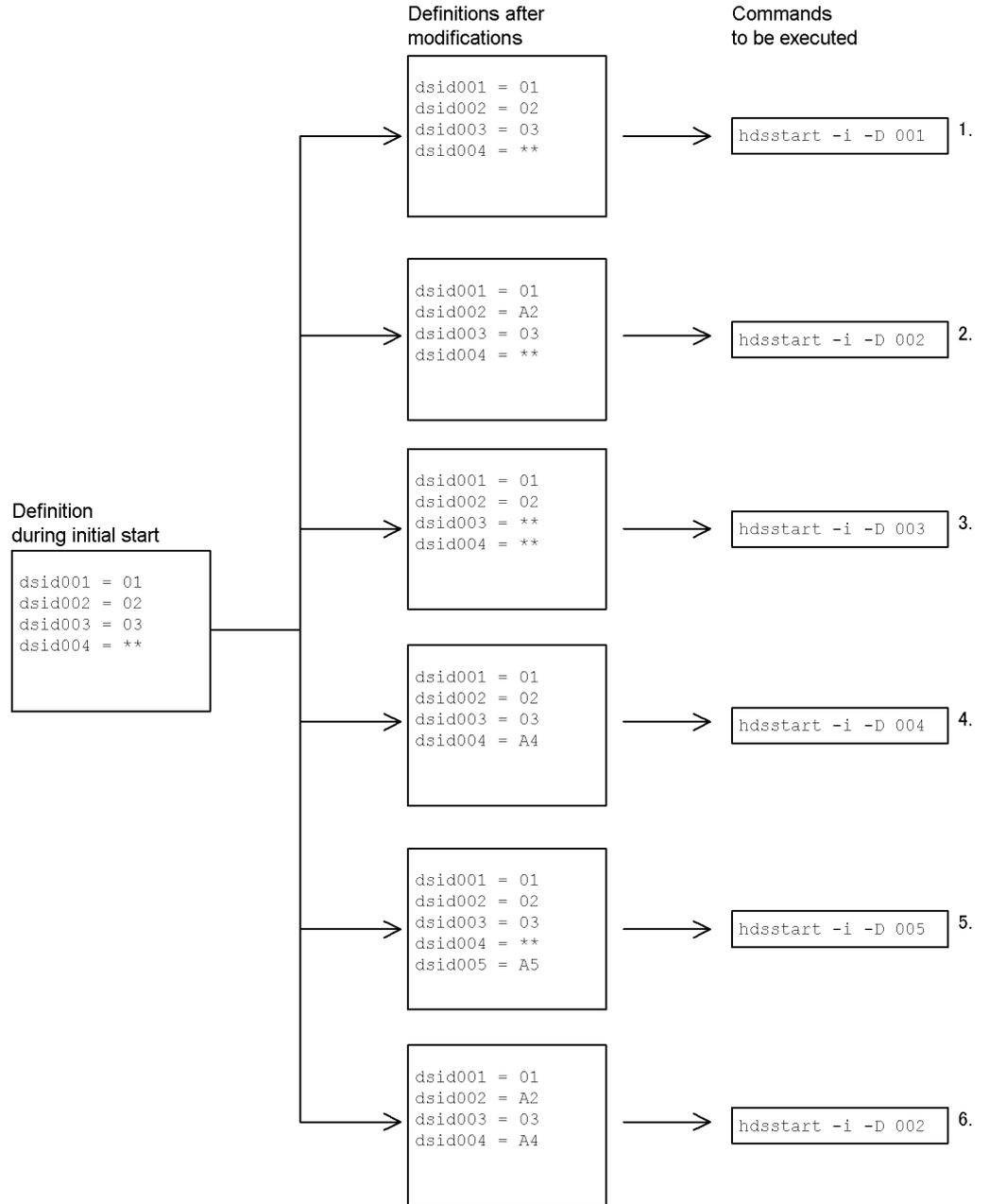
#2: If you execute the `hdsstart -i` command with `init` specified, the file is created with the specified size.

#3: Use an OS command to re-create this file.

Notes

- If an error occurs during partial initial start, you cannot execute normal startup or another partial initial start with a different data linkage identifier number specified while the error remains uncorrected. To correct a partial initial start error, eliminate the cause of the error, re-execute the partial initial start with the same data linkage identifier number specified, and then either terminate Datareplicator normally or execute initial start on the entire extraction environment.
- You can specify partial initial start only for data linkage identifier numbers that were defined at the time of the preceding initial start. You cannot specify a new data linkage identifier number that has been added since then. If the number of targets is expected to increase, specify `**` for future data linkage identifiers.

The following is a specification example of partial initial start:

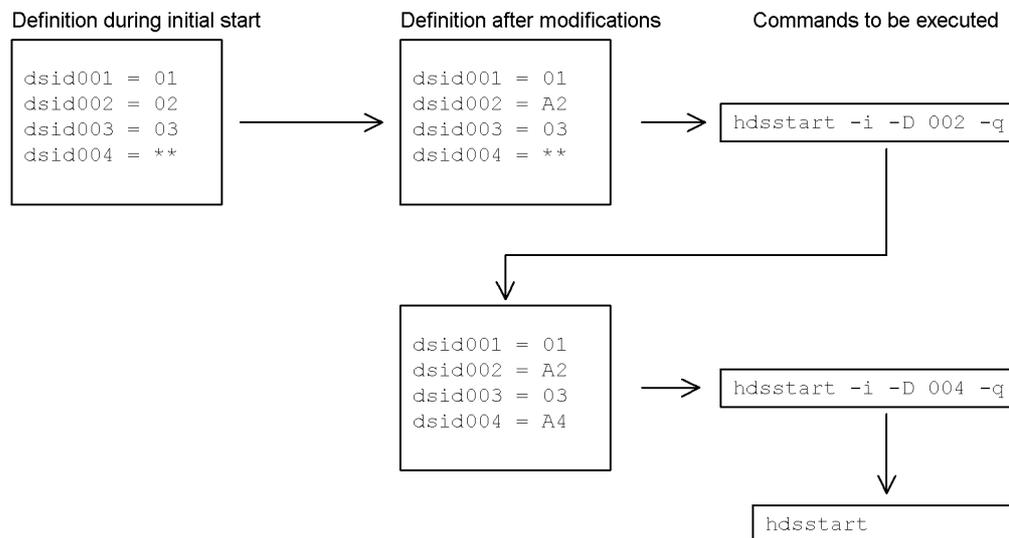


1. Executes partial initial start for data linkage identifier number 001 (data linkage

identifier 01).

2. Executes partial initial start for data linkage identifier number 002 (data linkage identifier A2) after modifying the definition.
3. Executes partial initial start for data linkage identifier number 003 (data linkage identifier **), which became an absent number after the definition was modified.
4. Executes partial initial start for data linkage identifier number 004 (data linkage identifier A4), which has been newly defined.
5. Results in a command error because data linkage identifier number 005 (data linkage identifier A5), which was added by modifying the definition, had not been defined at the time of the initial start.
6. Results in a command error because data linkage identifier number 004 (data linkage identifier A4) was modified, which is not the specified data linkage identifier number (data linkage identifier A2).

To execute partial initial start for more than one data linkage identifier, modify the definition and specify the `-q` option for one data linkage identifier at a time, and then execute partial initial start. After all partial initializations have been completed, execute normal start. The following is an example of the execution procedure:



- The following is the execution procedure for partial initial start:

1. Check that the target Datareplicator is stopped.
2. Use a text editor to modify the definitions.

3. Specify the data linkage identifier number subject to initialization in the `hdsstart -i -D` command in order to execute partial initial start.
 - If you use a system switchover configuration, you must activate the resources that are used by HiRDB Datareplicator (shared disk unit) prior to initialization or startup.

hdsstate (collect target Datareplicator status information)

Function

The `hdsstate` command outputs the current status of the target Datareplicator to the standard output. The command outputs the following information:

- Common information
Information common to all import processing for the target system.
- Synchronous import group information
Information about the synchronous import group.
- Import information queue file information
Information about the utilization status of the import information queue files.
- Reception processing information
Information about the status of reception processing, such as the protocol type and existence of reception events.
- Import processing information
Information about the status of import processing, such as the import processing mode and commit interval.

Format

```
hdsstate [ -d data-linkage-identifier | -g synchronous-import-group-name ]
```

Options

`-d` *data-linkage-identifier*

To collect target Datareplicator status information corresponding to a specified source system, specify the source system's data linkage identifier. Specifying ** for the data linkage identifier will result in an invalid-argument error.

For details about the data linkage identifier when the source database is a HiRDB, see *5.2 Extraction system definition*. For details about the data linkage identifier when the source database is a mainframe database, see the *VOS3 XDM/DS* manual.

If you omit the `-d` option, Datareplicator collects status information for all target Datareplicators corresponding to all source systems.

`-g` *synchronous-import-group-name*

Specifies the name of a synchronous import group in order to use the import transaction synchronization facility to collect status information for the data

linkage identifiers that constitute that synchronous import group.

If you omit the `-g` option, the command acquires status information for the target Datareplicators that corresponds to all source systems.

Rules

- If the `hdsstate` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.
- You can execute the `hdsstate` command only while the target Datareplicator is active.
- The target Datareplicator collects the status information as of the time the `hdsstate` command executes and outputs it to the standard output. Therefore, the output information might not match the actual status of the target Datareplicator.
- The target Datareplicator does not output information for a data linkage identifier specified as an absent number.

Output format

```

*****
**          HIRDB Datareplicator status information          **
**                      Thu Nov 10 13:47:41 2005          **
*****

----- Common information -----

hdsid = c0      shmid = 3504      semid = 907      1.
PID:   hdsmaster = 4692,      2.
       hdstcpmst = 4693,      hdsosimst = 4694

----- Sync-Group information -----

sync-group = Grp001      1.
shmid = 4      2.
semid = 1      3.
PID : hdsrefsync = 27266      4.
syncwait tran count = 11      5.

```

```

----- Replication information -----
dsid = a1 [sync-group:Grp001] .....1.

<EXTRACTED DATA QUEUE FILE INFORMATION>

wrap count = 1 .....2
qufile001 status : using .....3
qufile002 status : using
qufile003 status : using
qufile004 status : unused
qufile005 status : unused

max used ratio : 2005/11/10 13:47:41 70% .....4
current used ratio : 2005/11/10 13:47:41 10% .....5

<COMMUNICATION INFORMATION>

PID : hdsosimst = 4709 protocol = OSI .....6
receiving event = exist .....7
write position : qufile003, offset =2048 .....8

<REFLECTION INFORMATION>

PID : hdsdefserv = 4695 definition-shmid = 1305 .....9
reflection mode = tbl commit interval = 1 .....10

GROUP = AAAAA PID :hdsreflect = 4712, .....11.
                hdssqle = 4715
                status :data-wait1(last event = retbl) .....12.
                DB-status :normal .....13.

                read transaction = 4 .....14.
                read position : qufile003, offset = 1536 .....15.
                uocname = none .....16.

                Reflection count : .....17.
                ins = 4, upd = 0, del = 0, purge = 0,
                commit = 4, timestamp = 0
                commit(extract transaction) = 4
                reflect delay times = 0000:00:05/0000:10:00 .....18.

                SQL Distribute information : .....19.
                hdssqle[1]:
                ins = 2, upd = 0, del = 0, purge = 0,
                commit = 2, timestamp = 0
                hdssqle[2]:
                ins = 1, upd = 0, del = 0, purge = 0,
                commit = 1, timestamp = 0

GROUP = BBBBBB
-----
Import processing information for group BBBBBB
-----

dsid = a2 [sync-group:Grp001]

-----
Import processing status information for the source system with data
linkage identifier a2
-----

```

Common information

1. `hdsid to semid = 907`
`hdsid` displays the target Datareplicator identifier. `shmid` displays the shared memory ID. `semid` displays the semaphore ID.
2. `PID: hdsmaster to hdsosimst = 4694`
`hdsmaster` displays the import master process ID. `hdstcpmst` or `hdsosimst` displays the import communication master process ID corresponding to the specified communications protocol. If there is no import master process, an error results when the `hdsstate` command executes. If there is no import communications master process, 0 is displayed as the import communication master process ID.

Synchronous import group information

1. `sync-group=Grp001`
`sync-group` displays the synchronous import group name.
2. `shmid=4`
`shmid` displays the shared memory ID that is used for the synchronous import group.
3. `semid=1`
`semid` displays the semaphore ID that is used for the synchronous import group.
4. `PID: hdsrefsync=27266`
`hdsrefsync` displays the process ID of the synchronization managing process.
5. `syncwait tran count=11`
This item displays the number of target transactions that are being imported after the previous commit.

Import information**Import information file information**

1. `dsid = a1 [sync-group:Grp001]`
Displays the data linkage identifier specified in the `-d` option. If the specified data linkage identifier is not found, the `hdsstate` command results in an error.

For a data linkage identifier that is not specified in the synchronous import group, the output format is as follows:

dsid=xx

For a data linkage identifier that is specified in the synchronous import group, the output format is as follows:

dsid=xx [sync-group:synchronous-import-group-name]

2. wrap count = 1

Displays the number of times the import information queue file was wrapped since initial start of the target Datareplicator.

3. qufile001 to status: unused

Displays the utilization status for each import information queue file specified in the qufile001 to qufile008 operands in the import environment definition. The following are the utilization statuses:

using: In use (when update information is stored, the utilization status is always using)

unused: Unused

4. max used ratio:2005/11/10 13:47:41 70%

Displays the date and time the import information queue file's usage rate peaked (reached the maximum value) and what that peak usage rate was.

The following is the display format when the import information queue file has just been initialized, or when the maximum usage rate has just been reset by the hdschgstatus command:

```
max used ratio      : ****/**/** **:***:**      0%
```

5. current used ratio:2005/11/10 13:47:41 10%

Displays the date and time at which the hdsstate command was executed, and the import information queue file's usage rate at that time.

Reception processing information

6. PID: hdsosimst to protocol = OSI

hdsosimst displays the reception protocol ID. protocol displays one of following as the communications protocol:

TCP/IP: TCP/IP protocol

OSI: OSI protocol

If there is no reception process, -1 is displayed as the reception process ID.

7. `receiving event = exist`

Displays whether there is a reception event establishing connection with the source system:

`exist`: There is a reception event.

`not exist`: There is no reception event.

`N/A`: Not applicable (because the reception process is inactive).

8. `write position to offset = 2048`

Displays the write position of the update information. `qfilexxx` displays one of the `qfile001` to `qfile008` operands specified in the import environment definition that corresponds to the import information queue file in which the update information is written. `offset` displays the write position of the update information, expressed as the offset from the beginning of the file.

Import processing information

9. `PID: hdsdefserv to definition-shmid = 1305`

`hdsdefserv` displays the import definition server process ID.

`definition-shmid` displays the shared memory ID for storing the definition information.

If there is no import definition server process, -1 is displayed as the import definition server process ID.

10. `reflection mode to commit interval = 1`

`reflection mode` displays the import processing mode. `commit interval` displays the import processing commit interval. One of the following is displayed as the import processing mode:

`init`: Import processing is being initialized.

`trn`: Processing using the transaction-based import method is underway.

`tbl`: Processing using a table-based import method or using a UOC routine is underway.

`spd`: Import processing has been suspended (in this case, items 12 through 18 are not displayed).

`N/A`: Not applicable (because the definition server process is inactive).

For the import processing commit interval, the command displays the

interval at which commit is issued, expressed as a number of transactions at the source system. This is the value specified with the `cmtintvl`, `trncmtintvl`, or `tblcmtintvl` operand in the import environment definition or the value of the `-c` option in the `hdsrftl` command. Note that if the import process is engaged in startup processing, 0 is displayed.

11. `GROUP` to `hdssqle = 4715`

`GROUP` displays the name of an import group. `hdsreflect` displays the import process ID. `hdssqle` displays the import SQL process (UOC process) ID. Once import processing is stopped, -1 is displayed as the process ID.

The import group name to be displayed varies as follows:

`trngroup`: When reflection mode is `trn` or `init`

import-group-name: When reflection mode is `tbl`

`othergrp`: When no import group is specified

`uocxxx`: When a UOC routine is used (*xxx*: sequential number beginning with 001)

For the import SQL process (UOC process) ID, the command displays `hdssqle` for an import SQL process and the user-defined import UOC process name for an import UOC process. If there is no shared memory for storing definition information, the command displays `uoc [unknown]`.

12. `status` to `retbl`

`status` displays the status of import processing. `last event` displays the type of most recent event detected by the import processing.

The status of import processing is displayed as follows:

`run`: Executing

`definto-wait`: Waiting for definition information

`data-wait1`: Waiting for update information to be received (before importing the end-of-transmission message)

`data-wait2`: Waiting for update information to be received (after importing the end-of-transmission message and ready to execute the `hdsstop` command)

`event-stop`: Event stopped

`normal-stop`: Normal termination

`abnormal-stop`: Abnormal termination

`midway-stop`: Forced termination

The event type is displayed as follows:

`none`: No event detected

`trn`: Transaction-based import event

`tbl`: Table-based import event

`spd`: Import processing stop event

`retrn`: Transaction-based import restart event

`retbl`: Table-based import restart event

`cntreset`: Event to reset the import processing count

`end`: Termination event

`other`: Other

13. `DB-status: normal`

Displays the status of the target database as follows:

`normal`: Normal status or during initialization of import processing

`critical`: Import processing terminated abnormally during synchronization point processing on the import processing.

14. `read transaction = 4`

Displays the number of transactions read from the import information queue file since the beginning of import processing.

This is the total value for the update information that was stored in the import information queue file, regardless of commit. The number of committed transactions is displayed in item No. 17.

15. `read position to 1536`

Displays the read position of the update information. `qufilexxx` displays one of the `qufile001` to `qufile008` operands specified in the import environment definition that corresponds to the import information queue file from which update information is read. `offset` displays the read position of the update information, expressed as the offset from the beginning of the file.

16. `uocname = none`

If a UOC routine is used, `uocname` displays the name of the UOC routine. If no UOC routine is used, `none` is displayed. If there is no

shared memory for storing definition information, N/A is displayed.

17. Reflection count to commit(extract transaction) = 4

Displays the number of import operations since the beginning of import processing. For this item, the value that existed at the time of the previous session is inherited even when the target Datareplicator is restarted.

If a UOC routine is used for import processing, Reflection count displays the number of import information items passed to the UOC routine. If no UOC routine is used for import processing, the SQL execution counts are displayed as follows:

ins: Number of insert operations

upd: Number of update operations

del: Number of delete operations

purge: Number of purge operations

commit: Number of commits

timestamp: Number of insert operations on the time-ordered information table

The value is 0 if a UOC routine is used.

commit(extract transaction): Number of committed source transactions

18. reflect delay times = 0000:00:05/0000:10:00

Displays the amount of time that elapsed until update data was imported by the target HiRDB from the time it was committed at the source HiRDB; also displays the value specified in the import definition. The following table shows display formats:

Display format	Description
0000:00:00/ <i>defined-value</i>	Nothing has been output from an import process.
****:**:**/****:**:**	The value 0 was specified in the reflect_delay_limit_time operand in the import environment definition.
--:----:--/ <i>defined-value</i>	9999:59:59 was exceeded.
-xxxx:xx:xx/ <i>defined-value</i>	The amount of time that elapsed from when update was committed at the source RDAREA until it was imported is a negative value. This means that the source machine's time is ahead of the target machine's time.

19. SQL distribute information to timestamp = 0

Displays the number of import operations that have been committed per SQL import process. This value indicates the number of SQL processes that executed key range partitioning or hash partitioning, as shown below (*ins*, *upd*, *del*, *purge*, and *timestamp* correspond to those indicated in 17 above):

ins: Number of insert operations

upd: Number of update operations

del: Number of delete operations

purge: Number of purge operations

commit: Number of commits

timestamp: Number of insert operations on the time-ordered information table

Notes

- Common information items Nos. 1 and 2 are common to all replication nodes that belong to the target Datareplicator.
- Import information items Nos. 1-19 are retained for each data linkage identifier. When you omit the *-d* option in the *hdsstate* command, the command displays this information as many times as there are data linkage identifiers. When you specify the *-d* option, the command displays this information only for the specified data linkage identifiers.
- Import information items Nos. 9-19 are displayed as many times as there are import processing groups.
- Import information item No. 14 indicates the last transaction processed. No. 17 indicates the number of transactions whose import processing has been completed (committed to the target database).
- If Datareplicator is started by the *hdsstart -i* command and *reflection mode* is *spd*, the value 0 is displayed for *commit interval*.
- If data linkage uses update information in a SAM file, reception processing information is displayed as follows, in which case import information items Nos. 6-8 are not displayed:
 Communication process not effective
- Import information item No. 19 is displayed only when multiple SQL processes are started for a single import process, such as for key range partitioning or hash partitioning.
- The value displayed in item No. 2 in the import information is reset to 0 when it

exceeds 2,147,483,647.

- Import information items Nos. 14, 17, and 19 are reset to 0 once their values exceed 18,446,744,073,709,551,615.
- The information output in import information items Nos. 14, 17, and 19 is reset each time the target Datareplicator is started.
- The information output in import information items Nos. 17 and 19 is reset at the following times:
 - Initialization (including partial initialization) or start of the target Datareplicator[#]
 - Detection of an event specified in the `eventcntreset` operand in the import environment definition
- #: Applicable when `false` is specified in `reflect_counter_reset` in the import system definition
- The import processing count is reset in the following cases:
 - The target Datareplicator was started with a different import processing method than was used in the previous session.
 - The table-based import method was used and the following information has been changed for the import group from what it was in the previous session:
 - Import group name
 - Number of SQL processes
 - Partitioning method (key range or hash partitioning)
 - Number of key range groups when the key range partitioning method is used
 - Number of RDAREAs for a hash-partitioned table when the hash partitioning method is used

For details, see the `reflect_counter_reset` operand in the import system definition.

hdsstop (terminate the target Datareplicator)

Function

The `hdsstop` command terminates the target Datareplicator.

Format

```
hdsstop [ -t { event|immediate|force } | -q wait-time ]
        [ -w ]
```

Options

`-t { event|immediate|force }`

`-t event`

Specify this option to execute event termination.

`-t immediate`

Specify this option to execute immediate termination.

`-t force`

Specify this option to execute forced termination.

`-q wait-time`

~ <unsigned integer> ((5-600)) (seconds)

To terminate the reception process, and then terminate the target Datareplicator after importing all the data that has been received, specify this option with a reception process termination wait time.

Use this parameter only if the reception process cannot detect a line disconnection due to power failure or communications line failure at the source system, or if you want to forcibly execute normal termination on the target Datareplicator in order to achieve planned system switchover. Do not specify this option with the shell during normal operation (that is, when a transmission completion message is sent from the source system and the target Datareplicator is to be terminated after importing all the data that has been transmitted). Once you execute the `hdsstop` command with the `-q` option specified, you cannot specify any option other than `-t force` in any subsequent issuance of the `hdsstop` command.

Note, however, that, in any specific `hdsstop` command issuance, this parameter and the `-t` option are mutually exclusive.

`-w`

Specify this option to cause `hdsstop` command processing not to terminate after it sends a termination request to the target Datareplicator's master process

(hdsmaster) until after hdsmaster has terminated.

Specify this option when you need to terminate the command after the Datareplicator has terminated completely, such as when a Datareplicator termination shell is used for planned system switchover.

If the `-w` option is omitted, the `hdsstop` command terminates itself after sending a termination request to the master process (the master process and the command terminate asynchronously).

If the `-w` option is specified, the `hdsstop` command sends a termination request to the master process, waits until the master process terminates, and then terminates itself (when the command terminates, the master process has already terminated).

Rules

- If the `hdsstop` command terminates normally, it returns 0. If it terminates abnormally, it returns 1.
- You can use the `-t` option to select a desired termination mode. The following table shows the relationship between the `-t` option value and the termination mode.

Table 7-14: Relationship between the -t option value and the termination mode

Option	Termination mode	Termination processing
Not specified	Normal termination	After executing the <code>hdsstop</code> command, the target Datareplicator terminates when import processing through the source system's activity unit is completed. If the source database is a HiRDB, its activity unit is from normal start to normal termination of the entire source system (for details about the activity unit when the source database is a mainframe database, see the <i>VOS3 XDM/DS</i> manual). Depending on the status of the connection with the source system and the options specified in the <code>hdsstop</code> command, the target Datareplicator might result in normal termination in cases other than the above; for details, see <i>Table 7-15 Conditions of normal termination</i> .
<code>-t event</code>	Event termination	After executing the <code>hdsstop</code> command, the target Datareplicator terminates when import processing detects the first event sent from the source system. If the target Datareplicator completes processing through the source system's activity unit before detecting the event, it terminates immediately without waiting for the event. When the table-based import method is being used, import processing terminates when import of the last group is completed. Use this option only for executing planned termination; do not use this option to terminate the target Datareplicator immediately or after importing all information that has been received.

Option	Termination mode	Termination processing
-t immediate	Immediate termination	After executing the <code>hdsstop</code> command, the target Datareplicator terminates when it finishes importing the current update information being processed. When the table-based import method is being used, the target Datareplicator terminates the groups beginning with the one whose import processing has advanced the farthest, and then terminates itself when the last group has reached the termination point of the most advanced group. After execution of the <code>hdsstop</code> command, if the target Datareplicator receives new update information before the most advanced group has terminated, reception processing is terminated when the update information is stored in the import information queue file.
-t force	Forced termination	After executing the <code>hdsstop</code> command, the target Datareplicator terminates forcibly regardless of the import status of the update information. If synchronization point processing is underway, the target Datareplicator terminates upon completion of the synchronization point processing. Executing forced termination on the target Datareplicator might result in loss of conformity between the source and target databases. If you use the restart mode the next time you start the target Datareplicator, the target database will be restored automatically.

Table 7-15: Conditions of normal termination

Status of connection with the source system		Import status	Whether to execute normal termination
Connected ^{#1}		Import of all received information completed	--
		Import of all received information not completed	#3
Not connected	End-of-transmission message received (Normal termination at the source)	Import of all received information completed	Y
		Import of all received information not completed	#4
	End-of-transmission message not received (Forced termination at the source ^{#2})	Import of all received information completed	Y
		Import of all received information not completed	#4

Y: Normal termination is executed.

--: Normal termination is not executed.

#1: This includes the case where disconnection cannot be identified due to disabled keepalive feature, for example, even if the line is actually disconnected

because of a communications error.

#2: This includes detection of linkage cancellation log information by the `pdrplstop` command and communication line failure because of power interrupt at the source machine or a communications error, as well as forced termination at the source (`-t force`).

#3: You can execute normal termination forcibly with the `hdsstop` command with the `-q` option specified. The target Datareplicator forcibly closes the communication line and terminates itself when all received information has been imported. When the reception process cannot identify a line disconnection, use this termination method to forcibly execute normal termination.

#4: Once all unimported information has been imported, the import status is the same as *completed importing all received information*.

- If multiple source systems are subject to import processing, the target Datareplicator terminates after importing information from all such source systems. However, if the termination mode is set to forced termination, the target Datareplicator terminates immediately.
- If there is no transmission data from the source system and the update information has been imported through the end, the target Datareplicator terminates regardless of the `-t` option specification.

Notes

- During termination processing by the `hdsstop` command, the target Datareplicator does not receive connection requests from the source system.
- If the target system terminates abnormally after the `hdsstop` command has been entered but before termination processing is completed, the termination command's instruction accepted before the termination will be ignored.
- If import processing on all received update information has been completed and no new update information has been sent from the source system, executing the `hdsstop` command will terminate the target Datareplicator regardless of the specified command options. However, in the case of a `hdsstop` command with no options specified (normal termination), the target Datareplicator will be terminated only if the previous termination was normal termination.
- If you execute the `hdsstop` or `hdsrftcl` command more than once, the command's processing depends on the specified options. The following table shows the Datareplicator's action when the `hdsstop` or `hdsrftcl` command is executed more than once.

Table 7-16: Datareplicator's action when the hdsstop or hdsrfctl command is executed more than once

Second command executed		First command executed							
		hdsstop command with -t option specified				hdsrfctl command with -m option specified			
		None	event	immediate	force	trn	tbl	spd	immediate
hdsstop command with -t option specified	None	F	F	F	F	S	S	S	S
	event	F	F	F	F	S	S	S	S
	immediate	S	S	F	F	S	S	S	S
	force	S	S	S	F	S	S	S	S
hdsrfctl command with -m option specified	trn	F	F	F	F	F	F	D	D
	tbl	F	F	F	F	F	F	D	D
	spd	F	F	F	F	S	S	S	F
	immediate	F	F	F	F	S	S	S	S

S: Cancels the first command and starts processing the second command.

D: If the first command is completed, Datareplicator starts executing the second command; otherwise, an error results when the second command is executed and Datareplicator continues processing the first command.

F: An error results when the second command is executed and Datareplicator continues processing the first command.

- You cannot use the hdsstop command to terminate a target Datareplicator whose version is 05-02 or earlier when extraction processing has been terminated forcibly by the pdrplstop command or because communication between the source and target has been broken due to some kind of communications error. To terminate the target Datareplicator in such a case, execute the hdsstop -t immediate or hdsstop -t force command.

hdstrcredit (edit an activity trace file)

Function

The `hdstrcredit` command edits a trace file that has been output during source or target Datareplicator operation and outputs the edited contents to another file (or to the standard output). The command analyzes the entire contents of the activity trace file, and then outputs to the file.

If there are multiple errors based on the specified options, the command outputs as many error messages as it can check. Because the command stores the option analysis results in its local memory.

Format

```
hdstrcredit -f activity-trace-filename [ , activity-trace-filename ]
           [ -l { glbl | perf1 | perf2 } ]
           [ -p process-ID [ -p process-ID ] ... ] |
           [ -t start-date-and-time [ , end-date-and-time ] ]
           [ -o destination-filename ] [ -O shortfmt ]
```

Options

`-f activity-trace-filename [, activity-trace-filename]`

~ <character string of 1-255 bytes>

Specify the names of the activity trace files that are to be input to the `hdstrcredit` command. You can specify the following Datareplicator activity trace files:

Source Datareplicator:

Extraction master trace files and extraction node master trace files

Target Datareplicator:

Import trace files

A filename can be the relative or the absolute pathname. You can also use an environment variable, such as `$HDSPATH/filename`. If you specify a relative pathname, you must execute the command in the current directory.

You can specify no more than two files. If you specify two files, use the comma (,) as the delimiter and ensure that both files are in the same activity range (information collected during Datareplicator operation from the same start command to termination command). When you specify two files, do not specify any spaces before or after the delimiter comma. When two files are specified, the command compares their creation dates and begins by reading the one with the earlier creation date.

The following will result in a command error and processing will be cancelled:

- The `-f` option is omitted.
- The file specified in the `-f` option is not found (if two files are specified and only one of them exists, an error results).
- A specified value exceeds 255 bytes.
- The `-f` option is specified more than once.

```
-l { glbl |perf1 |perf2 }
~ <<glbl>>
```

Specify the type of information you want to output as a result of editing the trace file(s). Specify this value as lowercase letters; otherwise, an error results.

The following table shows the information that is output depending on the `-l` option specification:

Trace file information that is output	-l option value		
	glbl	perf1	perf2
Startup information Information indicating execution of processing during startup or termination. Used only once per normal process startup, this information indicates the start and termination of processing and its status.	Y	Y	Y
Global information Information collected at certain points during large-volume processing. This information is used periodically, such as for the unit of transfer or transactions, but its frequency is comparatively low.	Y	Y	Y
Local event information Information indicating changes in Datareplicator's internal processing. This information identifies changes in processing and is used for comparatively low-frequency processing whose volume is smaller than that of the global information.	Y	Y	Y
Performance information (1) Information collected over a comparatively large range (global information + some local events) for the purpose of performance analysis. This information is useful for determining overall performance.	--	Y	Y
Performance information (2) Performance analysis information collected over a smaller range, such as SQL1. When you choose to include this information, be aware that the amount of output can be very large. This provides almost complete performance information.	--	--	Y
Error information Information collected when errors occurred.	Y	Y	Y

Y: The information is output.

--: The information is not output.

Note that you must have specified the selected information for collection in the trace files. For details about the trace file information collection specification, see the source system definition or the target system definition.

The following will result in a command error and processing will be cancelled:

- Specifying a value that is not permitted.
- Specifying multiple options.

`-p process-ID [-p process-ID]`

~ <string of numeric characters>

Specify the process IDs for which trace file editing results are to be output. You can specify a maximum of eight process IDs in the format `-p process-ID -p process-ID . . .`

See the error information files for the process IDs. The command does not check the validity of a specified process ID. If a specified ID contains a nonnumeric character, the command assumes the string of numeric characters up to that point as the process ID.

If you specify more than eight process IDs, the command results in an error and cancels processing.

`-t start-date-and-time [, end-date-and-time]`

~ <YYYYMMDDHHMMSS>

Year (YYYY), month (MM), date (DD), hour (HH), minute (MM), second (SS)

<<all are subject to editing>>

Specify the dates and times defining a period that is to be subject to trace file editing and output. Specify each in the format year (4 digits), month (2 digits), date (2 digits), hour (2 digits), minute (2 digits), and second (2 digits). The specified range must be in the format `start-date-and-time [, end-date-and-time]`.

If you specify only a start date and time, all trace information collected after the specified date and time is edited. If you specify both start and end dates and times, all trace information collected between the specified dates and times is edited. Datareplicator does not check for a leap year or a leap second.

When you specify both start and end dates and times, do not specify any spaces before or after the delimiter comma. If any spaces are specified, Datareplicator will not recognize the specified information correctly.

The following will result in a command error and processing will be cancelled:

- The specification format is invalid.
- The *start-date-and-time* are the same as or later than the *end-date-and-time*.
- The `-t` option is specified more than once.

`-o destination-filename`

~ <character string of 1-225 bytes> <<standard output>>

Specify the name of the file to which the edited trace information is to be output. If you omit the `-o` option, Datareplicator outputs the information to the standard output. The specified file does not need to exist. If it already exists, the file is overwritten.

The filename can be the relative or the absolute pathname. You can also use an environment variable, such as `$HDSPATH/filename`. If you specify a relative pathname, you must execute the command in the current directory.

The following will result in a command error and the processing will be cancelled:

- The specified value exceeds 255 bytes.
- The `-o` option is specified more than once.

`-O shortfmt`

Specifies that the trace information is to be edited and output in the simple format of one entry per line. Datareplicator ignores any value other than `shortfmt` that is specified in the `-O` option (a warning message is issued).

Rules

To determine the command's execution results, check the command's return value or check for an error message. The following are the command's return values:

Return value	Termination status	Description
0	Normal termination	Datareplicator has finished reading and editing all data in the input trace files.
1	Abnormal termination	<ul style="list-style-type: none"> • An input trace file resulted in an I/O error; or, the edited results output operation resulted in an I/O error. • The contents of an input trace file were not valid. • A specified option was invalid and it was not possible to continue command processing.
Undefined	Abnormal termination	Signal interrupt occurred (<code>hdstrcredit</code> command does not handle signals).

Notes

- If an input activity trace file ends in the middle of a record, Datareplicator edits the file as follows:

When only one activity trace filename is specified in the `-f` option:

Datareplicator ignores the incomplete record and handles it as the end of the data.

When two activity trace filenames are specified in the `-f` option:

Datareplicator concatenates the incomplete record with the first record of the next file.

- If the OS returns an error (system call) during the record input operation, Datareplicator handles it as follows:

When the OS returns an error:

Datareplicator outputs an error message and terminates the command.

- To execute this command, use the user name that was used to initialize the source Datareplicator (`hdestart -i` command).

Output format

If the records collected in the activity trace file are normal, Datareplicator edits and outputs them. The following is an example of the results of trace file editing:

For a list of the process codes (, zzzz) displayed following the process ID (Pid), see Table 7-17 *Process codes*.

For the relationship between the value (zzzzzz) displayed as the identification name (Cpn) and the displayed information, see Table 7-18 *Relationship between the process code and identification name*.

The date and time are displayed in the format
4-digit-year-2-digit-month-2-digit-date
2-digit-hour:2-digit-minute:2-digit-second.6-digit-microsecond (the 2-digit hour is in 24-hour time). This format also applies to all subsequent date and time information.

3. Number of items Datareplicator was unable to send and additional character string information.

Add-strings displays a summary of the activity trace information. It is a character string of up to 24 bytes indicating the purpose of the activity trace, the status, and output source information (internal information).

4. Start time and end time

This information is displayed only for the performance-related information.

5. Target identifier (Sndid), data linkage identifier (Ujid), and data sequence ID (Seqn)

In the case of a target system, the target identifier is -----. This information is displayed only for the performance-related information.

6. Return information

This is a function's return information or information about SQL execution. In the case of a function's return information, the actual return value is displayed; in the case of an SQL execution, SQLCODE and SQLWARN6 or SQLWARN7 are output.

7. If there is any detail information in binary format, it is displayed in this format.

8. If there is any detail information in character string format, it is displayed in this format.

Additional-data provides details about the activity trace information. This is internal information and is referenced in the event of an error.

Table 7-17: Process codes

Process code	Process name
TRCR#	Trace information collection process
MSTR	Extraction master process or import master process
NMST	Extraction node master process
SNDR	Transmission process
SNDM	
SNDC	
CAPT	Extraction process
RCVT	Reception process (TCP/IP communication)
RCVO	Reception process (OSI communication)
DEFS	Import definition server process
RFCT	Import process
SQLF	Import SQL process
UOCG	Import UOC process

#: Output only once immediately after startup.

Table 7-18: Relationship between the process code and identification name

Process code	Value displayed for Cpn	Trace level	Description	Additional information that is output simultaneously
TRCR	QRCVST	GL	Start of activity trace collection	Message queue ID
MSTR	R_INIT	GL	INITENV (initialization) request issuance trace collection	Request type, object of request
	R_STRT	GL	START request issuance trace collection	Request type, object of request
	R_STOP	GL	STOP request issuance trace collection	Request type, object of request
	R_STTE	GL	STATE request issuance trace collection	Request type, object of request
	R_WTCH	GL	WATCH (monitoring) request issuance trace collection	Request type, object of request

Process code	Value displayed for Cpn	Trace level	Description	Additional information that is output simultaneously
NMST	R_INIT	GL	INITENV (initialization) request reception trace collection	N/A
	R_STRT	GL	START request reception trace collection	Request type, object of request
	R_STOP	GL	STOP request reception trace collection	Request type, object of request
	R_STTE	GL	STATE request reception trace collection	Request type, object of request
	R_WTCH	GL	WATCH (monitoring) request reception trace collection	Request type, object of request
SNDR	SNDSTR	ST	Start of transmission process	Displays hdesender start.
	SNDPPT	P1	Collection of port check transmission performance	Displays Sendperformance: portcheck send.
	SNDPDF	P1	Collection of extraction definition transmission performance	Displays Sendperformance: extract definition send.
	SNDPTW	P2	Collection of transaction management performance	Displays Sendperformance: transaction watch.
	SNDPEX	P1	Collection of update information transmission performance (transmission of all data during the transmission interval).	Displays Sendperformance: sendinterval.
	SNDEND	ST	Termination of transmission process	Displays hdesender stop.
	SNCPSD	P1	Collection of update information transmission performance (transmission of only one data segment)	Displays the transmission split type, number of transmitted transactions, number of update information items, and length in bytes.
	SNCPAU	P2	Collection of the ADT constructor information creation UOC performance	Displays ADT UOC call.

Process code	Value displayed for Cpn	Trace level	Description	Additional information that is output simultaneously
SNDR	SNCPLB	P2	Collection of the queue file storage format assembly performance	Displays EB Log.
	SNCPLC	P2	Collection of the queue file storage format assembly performance	Displays EC Log.
	SNCPUK	P2	Collection of the transmission checking UOC performance	Displays the UOC's return value.
	SNCPCV	P2	Collection of the send data editing performance (repetition column)	Displays hde_edt_extcnv2() call.
	SNCPCV	P2	Collection of the send data editing performance	Displays hde_edt_extcnv() call.
	SNDLYT	P1	Collection of the transmission delay period	Delay period
SNDM	SNMSTR	ST	Start of the extraction node master process	Displays hdesenmst start.
	SNMEND	ST	Termination of the extraction node master process	Displays hdesenmst end.
SNDK	SNPSTR	ST	Start of the transmission process	Displays hdesndprc start.
	SNPPEX	P1	Collection of the update information transmission performance (transmission of all data during the transmission interval)	Displays Sendperformance: sendinterval.
	SNPEND	ST	Termination of the transmission process	Displays hdesndprc end.
	SNPPPT	P1	Collection of the port check transmission performance	Displays Sendperformance: portcheck send.
	SNPPDF	P1	Collection of the extraction definition transmission performance	Displays Sendperformance: extract definition send.
	SNDLYT	P1	Collection of the transmission delay period	Delay period

Process code	Value displayed for Cpn	Trace level	Description	Additional information that is output simultaneously
CAPT	REDINI	ST	Collection of the READ start point immediately after the start of process	READ point information (binary format)
	SYNCQU SYNCRD	LC	Trace of READ and WRITE point synchronization processing	System log READ point and queue file WRITE point
	CMDTRC	LC	Trace of command acceptance	Details about the command request
	CPDLYT	P1	Collection of the extraction delay period	Delay period
RCVT	RCVDP1	P1	Data reception	Data reception status
	TCPSOP	ER	socket open error	Function's return value, errno
	TCPSOP	GL	Termination of socket open	Function's return value
	TCPSBN	ER	bind error	Function's return value, errno
	TCPSBN	GL	Termination of bind	Function's return value
	TCPSLI	GL	Result of listen	Function's return value
	TCPSCP	GL	Wait for accept	N/A
	TCPSCP	GL	accept return	Function's return value, errno
	TCPFRK	GL	fork error	Function's return value, errno

Process code	Value displayed for Cpn	Trace level	Description	Additional information that is output simultaneously
RCVO	OSITOP	GL	Result of <code>t_open</code>	Function's return value, <code>errno</code>
	OSITBN	GL	Termination of <code>t_bind</code>	Function's return value, <code>errno</code>
	OSITAL	GL	Termination of <code>t_alloc</code>	Function's return value, <code>errno</code>
	OSITLI	GL	Wait for <code>t_listen</code>	N/A
	OSITLI	GL	Result of <code>t_listen</code>	Function's return value, <code>errno</code>
	OSITOP	GL	Result of <code>t_open</code>	Function's return value, <code>errno</code>
	OSITBN	GL	Result of <code>t_bind</code>	Function's return value, <code>errno</code>
	OSITAC	GL	Result of <code>t_accept</code>	Function's return value, <code>errno</code>
	OSIFRK	GL	Result of <code>fork</code>	Function's return value, <code>errno</code>
DEFS	DFINIT	ST	Start of process	Data linkage identifier
	DFEXIT	ST	Termination of process	N/A
	DEFMKP	LE	Start of subprocess (INIT)	Event code, process ID
	DEFMKP	LC	Start of subprocess (NORMAL)	Event code, process ID
	DEFMKP	LC	Start of subprocess (RERUN)	Event code, process ID
	DEFCOM	LC	Detection of command entry	Command name
	DEFMKP	LC	Start of subprocess (event)	Event code, process ID
	DEFMKP	LC	Start of subprocess (<code>hdsstop</code>)	Event code, process ID

Process code	Value displayed for Cpn	Trace level	Description	Additional information that is output simultaneously
RFCT	RFINIT	ST	Start of import process	Datareplicator identifier, data linkage identifier
	EVSTOP	GL	Termination due to the previous event termination	Event ID
	IMSTOP	GL	Immediate termination during start processing	N/A
	RFSTRT	GL	Start of import processing	Start mode, queue file offset
	RFSIGS	LE	Signal reception	Received signal
	RFQGET	P1	Detection of end of queue file	Queue file offset
	RFQGET	P1	Input of queue file data	Queue file offset, update information (first 128 bytes)
	RFTRNC	P1	Transaction entry in the update information	Transaction counter, number of update information items in transaction
	EVENTC	LE	Event detection	Event ID, process status code 1, 2
	RFTERM	ST	Termination of import process	Termination mode, queue file offset
	RFPIPR	P2	Reception of SQL execution result	Destination entry, read size or return code
	RFPIPW	P2	Transmission of SQL execution instruction to the SQL process or UOC process	Destination entry, write size, SQL assembly information (first 128 bytes)
RFRCVR	P1	Transaction recovery result during rerun (2-phase commit only)	Return code	
RFCT	RFCOMT	P1	Result of COMMIT	Reason for commit, return code
	RFENDC	P1	Result of END/DISCONNECT	Type, return code
	RFEXIT	ST	Termination of import process	Type, return code
	RFDLTY	P1	Collection of the import delay period	Delay period

Process code	Value displayed for Cpn	Trace level	Description	Additional information that is output simultaneously
SQLE	SQINIT	ST	Start of process	Start mode (dsid, group name, segment number)
	SQEXIT	ST	Termination of process	Termination mode (NORMAL, FORCE, ERROR)
	SQREQ	GL	Acceptance of termination request	N/A
	XARERN	LC	Recovery execution (2-phase commit method)	Recovery type
	TRETRY	LC	Transaction retry	N/A
	CNCT_S	P2	Before execution of connect	N/A
	CNCT_E	P2	After execution of connect	SQLCODE
	PINS_S	P2	Before execution of prepare (insert)	SQL statement (127 bytes maximum)
	PINS_E	P2	After execution of prepare (insert)	SQLCODE
	PUPD_S	P2	Before execution of prepare (update)	SQL statement (127 bytes maximum)
	PUPD_E	P2	After execution of prepare (update)	SQLCODE
	PDEL_S	P2	Before execution of prepare (delete)	SQL statement (127 bytes maximum)
	PDEL_E	P2	After execution of prepare (delete)	SQLCODE
	EINS_S	P2	Before execution of execute (insert)	Table name
	EINS_E	P2	After execution of execute (insert)	SQLCODE
	EUPD_S	P2	Before execution of execute (update)	Table name
	EUPD_E	P2	After execution of execute (update)	SQLCODE

Process code	Value displayed for Cpn	Trace level	Description	Additional information that is output simultaneously
SQLE	EDEL_S	P2	Before execution of execute (delete)	Table name
	EDEL_E	P2	After execution of execute (delete)	SQLCODE
	EXEI_S	P2	Before execution of execute immediate	SQL statement (127 bytes maximum)
	EXEI_E	P2	After execution of execute immediate	SQLCODE
	CMIT_S	P2	Before execution of commit	N/A
	CMIT_E	P2	After execution of commit	SQLCODE
	RLBK_S	P2	Before execution of rollback	N/A
	RLBK_E	P2	After execution of rollback	SQLCODE
	XAOP_S	P2	Before execution of xa_open()	N/A
	XAOP_E	P2	After execution of xa_open()	Return value of xa_open()
	XAPR_S	P2	Before execution of xa_prepare()	N/A
	XAPR_E	P2	After execution of xa_prepare()	Return value of xa_prepare()
	XACM_S	P2	Before execution of xa_commit()	N/A
	XACM_E	P2	After execution of xa_commit()	Return value of xa_commit()
	XARL_S	P2	Before execution of xa_rollback()	N/A
	XARL_E	P2	After execution of xa_rollback()	Return value of xa_rollback()
	DISC_S	P2	Before execution of disconnect	N/A
	DISC_E	P2	After execution of disconnect	SQLCODE
	SQPIPR	P2	Read update information	Request type, read size
	PWRT_A	P2	Transmission of PRE-C result	Execution result, write size
SQPIPW	P2	Transmission of COMMIT result	Execution result, write size	

Process code	Value displayed for Cpn	Trace level	Description	Additional information that is output simultaneously
SQLE	CUOC_S	P2	Before execution of column UOC	Column UOC function type
	CUOC_E	P2	After execution of column UOC	Column UOC function type, column UOC function execution result
	CUOC_S	P2	Before execution of column UOC	Column UOC function type
	CUOC_E	P2	After execution of column UOC	Column UOC function type, column UOC function execution result
	SQPIPW	P2	Transmission of update result (up to the synchronization point)	Execution result, write size
UOCG	UCINIT	ST	Start of process	Start mode
	UCEXIT	ST	Termination of process	Termination mode (NORMAL, FORCE, ERROR)
	UBE1_S	GL	Call to hds_ubegin()	N/A
	UEDT_S	GL	Call to hds_uedit()	N/A
	UEND_S	GL	Call to hds_uend()	N/A
	UCPIPR	P2	Read SQL execution instruction	N/A
	UBE1_E	P2	Return from hds_ubegin()	N/A
	UEDT_E	P2	Return from hds_uedit()	N/A
	UEND_E	P2	Return from hds_uend()	N/A
	UCPIPW	P2	Transmission of update result (up to the synchronization point)	N/A
	UCERREQ	GL	Acceptance of termination request	N/A

Output example

The following is an example of an activity trace collected at a target Datareplicator. This example collected the activity trace under the following conditions:

- `int_trc_lvl=p1` was specified in the target system definition.
- `hdstrcredit -f reftrc.trc1 -l perf1 -o reftrc.res` was executed.


```

No.00000027 Lvl:LE Pid:02897,RFCT Cpn:RFSIGS Date:2002-03-29 17:47:31.089582
  Unsendable-count:00000 Add-strings:Recieve Signal=01
No.00000028 Lvl:LE Pid:02776,DEFS Cpn:DEFCOM Date:2002-03-29 17:47:31.091007
  Unsendable-count:00000 Add-strings:Command accept
  Additional-data:
    command = hdsstop(normal)
No.00000029 Lvl:P1 Pid:02897,RFCT Cpn:RFQGET Date:2002-03-29 17:47:31.091196
  Unsendable-count:00000 Add-strings:Ofst(En)=0,2560
  Sndid:----- Ujid:bf Seqn:ffffffffffffffffffffffffffffffffffff
  Additional-data:
    00000040 e2200000 *...@. .. *
No.00000030 Lvl:P1 Pid:02897,RFCT Cpn:RFQGET Date:2002-03-29 17:47:31.112715
  Unsendable-count:00000 Add-strings:No data in QUEUE
  Additional-data:
    Queue offset=0,3072
No.00000031 Lvl:ST Pid:02897,RFCT Cpn:RFTERM Date:2002-03-29 17:47:31.144381
  Unsendable-count:00000 Add-strings:Term mode=Normal
  Additional-data:
    Queue file position = [0, 2560]
No.00000032 Lvl:P1 Pid:02897,RFCT Cpn:RFENDC Date:2002-03-29 17:47:31.144612
  Unsendable-count:00000 Add-strings:End Req,Retrn=1
  Sndid:----- Ujid:bf Seqn:ffffffffffffffffffffffffffffffffffff
No.00000033 Lvl:GL Pid:02898,SQLE Cpn:SQREQ Date:2002-03-29 17:47:31.144853
  Unsendable-count:00000 Add-strings:END request accept
No.00000034 Lvl:ST Pid:02898,SQLE Cpn:SQEXIT Date:2002-03-29 17:47:31.153876
  Unsendable-count:00000 Add-strings:SQLE process end
  Additional-data:
    exit code=1
No.00000035 Lvl:ST Pid:02897,RFCT Cpn:RFEXIT Date:2002-03-29 17:47:31.176136
  Unsendable-count:00000 Add-strings:RFCT-PROC EXIT,RTN=0
No.00000036 Lvl:ST Pid:02776,DEFS Cpn:DFEXIT Date:2002-03-29 17:47:32.204934
  Unsendable-count:00000 Add-strings:Defserv process end

```

No. 00000002, 00000007, 00000014, 00000018, and so on

Each of these processes was started by import processing.

No. 00000003, 00000004, 00000025, and 00000026

Reception processing was executed. Perf-info displays the time reception processing started and the time data reception was completed.

No. 00000016 and 00000019

Times at which data was read from the import queue file and time at which the transaction was completed (COMMIT). The records between identification name (Cpn) RFQGET, which indicates the import queue file read operation, and RFCOMT, which indicates commit, are the range of this import processing transaction. The interval between these times is the amount of transaction processing time at the target system.

No. 00000031 to the end

Each process's termination processing at the target system. For each process ID, the time at which processing terminated is shown.

If you specify p2 for the activity trace collection level and -1 perf2 in the

hdstrcredit command, each SQL statement's issuance and completion time is displayed in addition to the information shown above.

pdlogchg (modify the status of log-related files)

Function

The `pdlogchg` command forcibly places a specified file group in unload completed status or in the source Datareplicator's extraction completed status.

Format

```
pdlogchg -d sys [ -s server-name ] -g file-group-name [ -R ]
```

Options

This section explains only the option that can be specified when data linkage is used with the source Datareplicator. For details about the other options, see the *HiRDB Version 8 Command Reference* manual.

-R

When HiRDB Datareplicator linkage is used, this option forcibly changes the specified file group from extracting status to extraction completed status.

When this option is specified, Datareplicator changes only the extraction status from extracting to extraction completed. When this option is omitted, Datareplicator only changes the unload status from unload wait to unload completed.

pdls (display the HiRDB system status)

This is a HiRDB command. For details about the rules and notes, see the manual *HiRDB Version 9 Command Reference*.

Function

The `pdls` command displays the status of a HiRDB system feature.

Format

```
pdls -d rpl [ -j ][ -u unit-identifier ][ -s server-name ]
```

Options

`-d feature-subject-to-status-display`

~ <<svr>>

Specify the feature whose status is to be displayed.

`rpl`

Display the status of HiRDB Datareplicator linkage.

This option displays whether HiRDB Datareplicator linkage is being used. It also displays the source Datareplicator's status of extracting system log information from the system log file.

`-j`

Specify this option to display the source Datareplicator's status of extracting system log information from the system log file. When this option is specified, the command displays the following information:

- Whether the HiRDB Datareplicator linkage facility is being used
- The source Datareplicator's status of extracting system log information from the system log file
- The source HiRDB's status of outputting system log information to the system log file

When this option is omitted, the command displays only whether the HiRDB Datareplicator linkage facility is being used; it does not display the system log extraction status or output status.

`-u unit-identifier`

~ <identifier of 4 characters>

To display the source Datareplicator's system log extraction status at a specific unit, specify the unit identifier.

pdls (display the HiRDB system status)

When you omit the `-u` option, the status is displayed for all servers in the system. If you omit the `-j` option and specify the `-u` option, the command displays only whether the system and this unit are using the HiRDB Datareplicator linkage facility.

`-s server-name`

~ <identifier of 1-8 characters>

To display the source Datareplicator's system log information extraction status at a specific server, specify the server name.

When you omit the `-s` option, the status is displayed for all servers in the system. If you omit the `-j` option and specify the `-s` option, the command displays only whether the system and the unit containing this server are using the HiRDB Datareplicator linkage facility.

pdrplstart (start HiRDB Datareplicator linkage)

This is a HiRDB command. For details about the rules and notes, see the manual *HiRDB Version 9 Command Reference*.

Function

The `pdrplstart` command starts HiRDB Datareplicator linkage.

Format

```
pdrplstart
```

pdrplstop (cancel HiRDB Datareplicator linkage)

This is a HiRDB command. For details about the rules and notes, see the manual *HiRDB Version 9 Command Reference*.

Function

The `pdrplstop` command cancels HiRDB Datareplicator linkage.

Note:

Use this command only to cancel data linkage using HiRDB Datareplicator. Do not use this command to temporarily suspend data linkage. If `pdrplstop` is executed, inconsistencies occur between the source and target databases.

Format

`pdrplstop [-f]`

Options

`-f`

Specify this option to forcibly terminate HiRDB Datareplicator linkage regardless of the HiRDB Datareplicator's system log extraction status. When this option is specified, the command terminates HiRDB Datareplicator linkage even if termination processing is underway at a unit.

Chapter

8. User Own Coding Routines

You can augment Datareplicator's data linkage applications by creating user own coding (UOC) routines. This chapter provides an overview of the types of UOC routines supported by Datareplicator and explains the creation and function syntax of the available UOC routines; it also provides coding examples.

- 8.1 Import information editing UOC routine
- 8.2 Column data editing UOC routine
- 8.3 Send data UOC routine

8.1 Import information editing UOC routine

An *import information editing UOC routine* that you create can be used to edit any update information and import it into a target database. The target Datareplicator will use the table-based import method to execute the import processing.

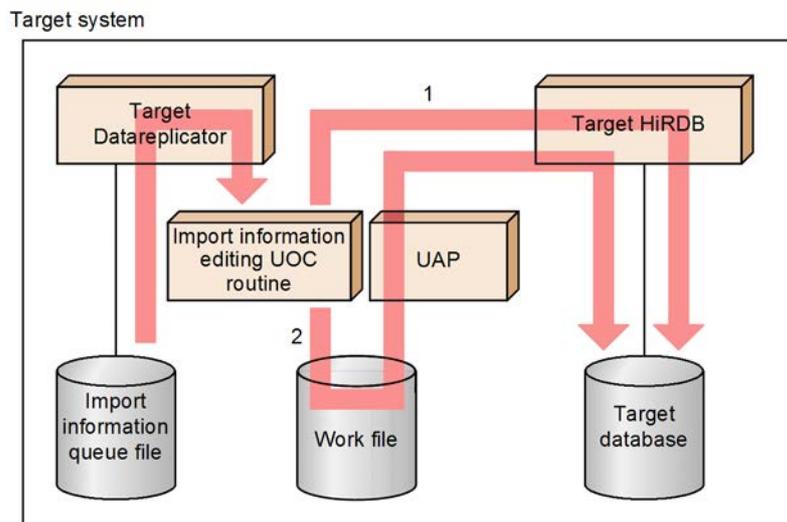
8.1.1 Overview of an import information editing UOC routine

There are two ways to use an import information editing UOC routine to import update information into a target database:

- By issuing SQL statements
- By output to a work file

The following figure shows the procedure for importing update information.

Figure 8-1: Procedure for importing update information using an import information editing UOC routine



1. Issuing SQL statements

With this method, the import information editing UOC routine issues SQL statements based on the update information received from the target Datareplicator, and then imports the update information into the target database.

2. Output to a work file

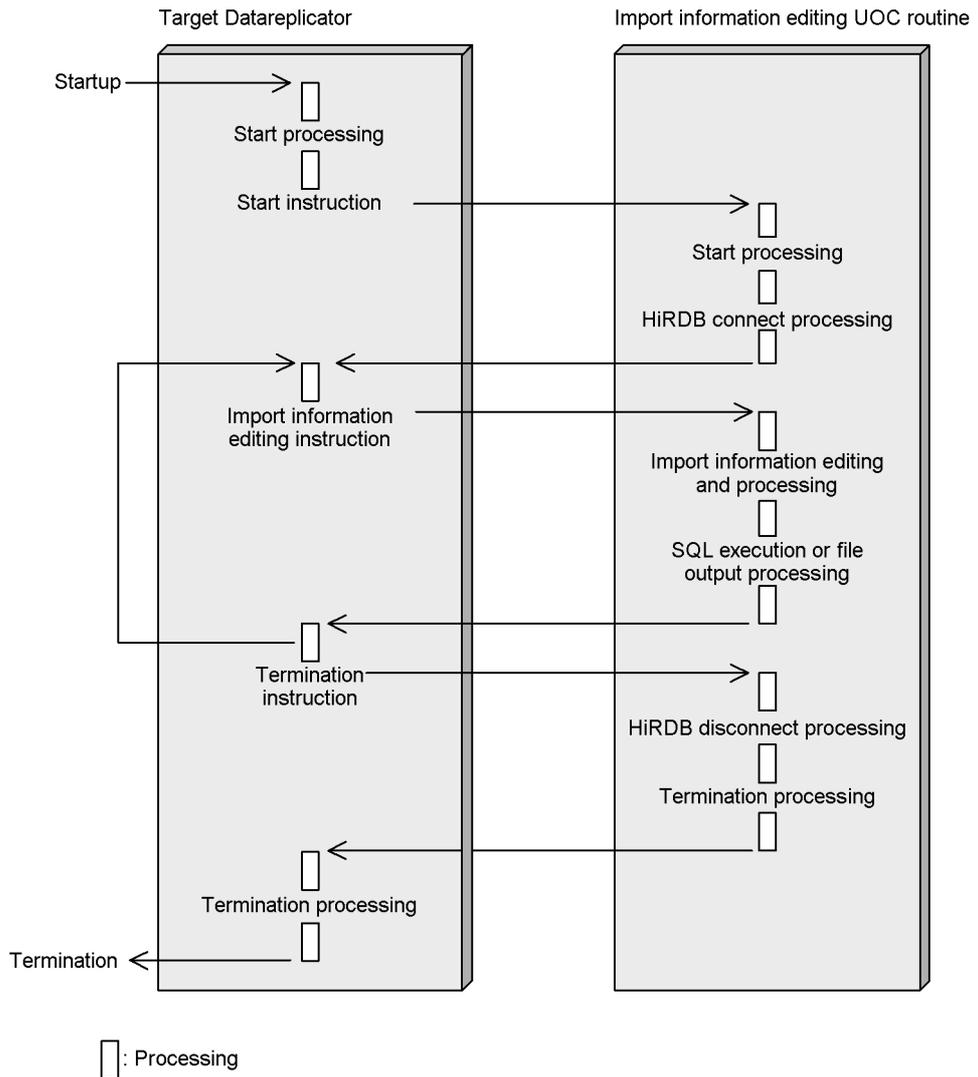
With this method, the import information editing UOC routine edits update information received from the target Datareplicator and outputs it to a work file.

It then uses a UAP to retrieve the update information from the work file, and imports it into the target database.

(1) Flow of control during processing by an import information editing UOC routine

The following figure shows the flow of control during processing by an import information editing UOC routine and the target Datareplicator.

Figure 8-2: Flow of control during processing by an import information editing UOC routine



Start instruction

When the target Datareplicator is started, a start instruction is issued to the UOC routine as an extend starting import processing (call to UOC's `hds_ubegin()`).

Import information editing instruction

The import process reads the update information sent from the source system and issues an update information editing instruction for each update information item that has been read (call to UOC's `hds_uedit()`).

Termination instruction

When the target Datareplicator is terminated, a termination instruction is issued to the UOC routine as an extension of terminating the import processing (call to UOC's `hds_uend()`).

The functions of the import information editing UOC are called at the following times:

- `hds_ubegin()`

This function is called only when import processing is started (when the import process is started by the `hdsstart` or `hdsrfctl` command or is restarted by detection of a definition change[#] or mode change event).

- `hds_uedit()`

This function is called for each update information item.

- `hds_uend()`

This function is called only when import processing is terminated (when the import process is terminated by the `hdsstop` or `hdsrfctl` command or is terminated temporarily by detection of a definition change[#] or mode change event).

[#]: As for a detection of a definition change, if selection definitions are used in XDM/DS, the definitions are sent each time connection is re-established with the target system and `hds_ubegin()` and `hds_uend()` are called, even when no change has been made to the definitions.

The following shows the number of executions of each function during normal processing (from start to termination of the target Datareplicator), where n is the number of updates:

- `hds_ubegin()`: 1
- `hds_uedit()`: 1 to n
- `hds_uend()`: 1

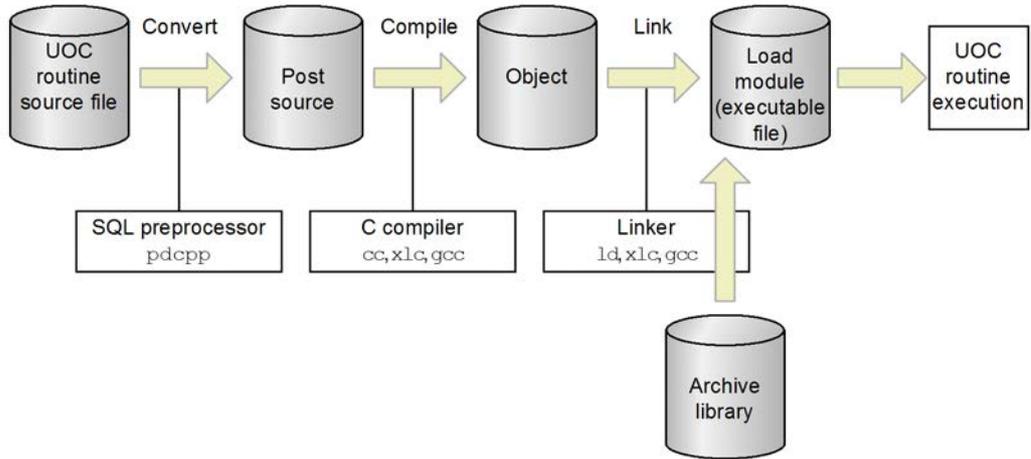
(2) Units in which update information is passed

Update information is passed from the target Datareplicator to the import information editing UOC routine in units of SQL executions. A `COMMIT` request for synchronization point processing is passed individually.

8.1.2 Creating an import information editing UOC routine (UNIX)

This section explains the procedure for creating an import information editing UOC routine with UNIX Datareplicator. If your import information editing UOC routine will use SQL statements, you cannot complete it directly. Instead, you must first convert it to the source format supported by the C compiler, and then compile it. The following figure shows the procedure up to the point where an import information editing UOC routine is executed.

Figure 8-3: Import information editing UOC routine's execution procedure



The type of load module for an import information editing UOC routine that is created depends on the type of Datareplicator, as shown in the following table.

Type of Datareplicator		Type of load module for an import information editing UOC routine
32-bit edition	HP-UX edition Solaris edition AIX 5L edition Linux edition	32-bit edition ^{#1}
64-bit edition	HP-UX edition Solaris edition AIX 5L edition	32-bit edition ^{#1}
IPF version	HP-UX edition	32-bit edition ^{#1}
		64-bit edition ^{#2}
	Linux edition	64-bit edition ^{#2}

#1

Code a UOC routine that runs in the 32-bit mode.

#2

Code a UOC routine that runs in the 64-bit mode. The following notes apply to this coding:

- Because the `long` type is recognized as an 8-byte integer, change the `long` type to the `int` type.
- A pointer has a length of eight bytes. There is no need to modify the pointer length unless the UOC routine contains processes in which the pointer length is important for processing.
- The functions used when an import information editing UOC routine is created might differ from those used in the 32-bit mode. Check if all functions are supported in the 64-bit mode.
- Because declaration of embedded variables using the `long` type is not supported, you must change `long` type embedded variables to the `int` type.

(1) Creating the interface

To create a shared library for an import information editing UOC routine interface:

1. Select a target column that is to use the import information editing UOC routine interface.
2. Create the function for the import information editing UOC routine interface.
3. If SQL statements are used in the import information editing UOC routine, use the target HiRDB's SQL preprocessor to generate source files that can be compiled by a C compiler.

For details about the SQL preprocessor, see the *HiRDB Version 9 UAP Development Guide*.

4. Compile and link the functions for the import information editing UOC routine interface.

(2) Compiling and linking

The compiling and linking procedures are explained for each OS.

If an unresolved link error occurs, specify all libraries required during linking.

(a) In HP-UX

- Compilation of a 32-bit edition UOC routine in an environment for the IPF version

```
cc -c +DD32 -I /HiRDB/include -I /opt/hirdbds/include/32 UOC-source-file-name
```

- Compilation of a 64-bit edition UOC routine in an environment for the IPF version

```
cc -c +DD64 -I /HiRDB/include -I /opt/hirdbds/include UOC-source-file-name
```

- Compilation of a UOC routine in an environment for the 32-bit or 64-bit edition

```
cc -c -I /HiRDB/include -I /opt/hirdbds/include UOC-source-file-name
```

-c:

Object creation option

-I /HiRDB/include:

For the underlined part, specify the target HiRDB installation directory.

-I /opt/hirdbds/include:

Directory that stores the header files provided by Datareplicator (always /opt/hirdbds/include). You can specify multiple directories if the UOC routine requires other header files.

UOC-source-file-name:

Name for the source file that will be obtained from the SQL preprocessor in a format supported by the C compiler

For details about the syntax of the compile command, see the applicable OS reference manual.

- Linkage of a 32-bit edition UOC routine in an environment for the IPF version

```
ld -o UOC-routine-name -u main UOC-routine-object-name
/opt/hirdbds/lib/32/libhdsuif.a /opt/hirdbds/lib/32/libhdsulb.so
/opt/hirdbds/lib/32/libhdscom.so /opt/hirdbds/lib/32/libhdspsc.so -lc#
```

- Linkage of a 64-bit edition UOC routine in an environment for the IPF version

```
ld -o UOC-routine-name -u main UOC-routine-object-name
/opt/hirdbds/lib/libhdsuif.a /opt/hirdbds/lib/libhdsulb.so
/opt/hirdbds/lib/libhdscom.so /opt/hirdbds/lib/libhdspsc.so -lc#
```

- Linkage of a UOC routine in an environment for the 32-bit or 64-bit edition

```
ld -o UOC-routine-name
-u main UOC-routine-object-name /opt/hirdbds/lib/libhdsuif.a
/opt/hirdbds/lib/libhdsulb.s1
/opt/hirdbds/lib/libhdscom.s1
/lib/crt0.o -lc#
```

Notes:

1. Specify for *UOC-routine-name* a name for the executable file; specify this same filename in the import definition. Before executing the UOC routine, you must specify in the PATH environment variable the pathname of the directory containing the executable file. Specify for *UOC-routine-object-name* the name of the object file obtained from the compilation process. The libraries under /opt/hirdbds/lib are provided by the target Datareplicator; you must specify these libraries.
2. The name of the executable file specified in the linkage command cannot begin with hds or duplicate any name reserved by any program already installed in the system.

#

To execute SQL statements from within the UOC routine, specify the following library at the end of the command:

- Creating a 32-bit edition load module using the IPF version

```
+s -L HiRDB-installation-directory/client/lib -lzcltk
```

- Creating a 64-bit edition load module using the IPF version

```
+s -L HiRDB-installation-directory/client/lib -lzcltk64
```

- 32-bit or 64-bit edition

```
+s -L HiRDB-installation-directory/client/lib -lzclt
```

+s means that the target HiRDB's installation directory is to be used separately from the target HiRDB directory; when you specify +s, also specify the SHLIB_PATH environment variable. -L indicates the default search path when SHLIB_PATH is omitted. -l specifies the target HiRDB's linkage library name.

(b) In Solaris

- UOC routine compilation

```
cc -c -I /HiRDB/include -I /opt/hirdbds/include UOC-source-file-name
```

-c:

Object creation option

-I /HiRDB/include:

For the underlined part, specify the target HiRDB installation directory.

-I /opt/hirdbds/include:

Directory that stores the header files provided by Datareplicator (always /opt/hirdbds/include). You can specify multiple directories if the UOC routine requires other header files.

UOC-source-file-name:

Name for the source file that will be obtained from the SQL preprocessor in a format supported by the C compiler

- UOC routine linkage

```
/opt/SUNWspro/bin/cc -o UOC-routine-name -u main
```

UOC-routine-object-name

```
/opt/hirdbds/lib/libhdsuif.a
```

```
/opt/hirdbds/lib/libhdscom.so
```

```
/opt/hirdbds/lib/libhdsulb.so
```

```
/opt/hirdbds/lib/libhdspck.so
```

```
-L/HiRDB/libl -R /HiRDB/lib -lzclt -lsocket -lnsl -lc  
-ldl
```

Note 1:

The `-L` option line contains the libraries that are used for executing SQL statements from within the UOC routine. For the underlined part, specify the target HiRDB installation directory.

Note 2:

You can omit the `-R` option. When you omit the `-R` option, you must add `HiRDB-installation-directory/lib` to `LD_LIBRARY_PATH`.

(c) In AIX

- UOC routine compilation

```
xlc -c -I HiRDB/include -I /opt/hirdbds/include UOC-source-file-name
```

`-c:`

Object creation option

`-I HiRDB/include:`

For the underlined part, specify the target HiRDB installation directory.

`-I /opt/hirdbds/include:`

Directory that stores the header files provided by Datareplicator (always `/opt/hirdbds/include`). You can specify multiple directories if the UOC routine requires other header files.

UOC-source-file-name:

Name for the source file that will be obtained from the SQL preprocessor in a format supported by the C compiler

For details about the syntax of the compile command, see the applicable OS reference manual.

- UOC routine linkage

```
xlc -o UOC-routine-name -u main UOC-routine-object-name
/opt/hirdbds/lib/libhdspsc.a /opt/hirdbds/lib/libhdsulb.a
/opt/hirdbds/lib/libhdscom.a
-L HiRDB/client/lib -Wl,-blibpath:./usr/lib:/lib:HiRDB/client/lib -lzclt
/opt/hirdbds/lib/libhdsuif.a -lm -lc
```

Note 1:

For *UOC-routine-name*, specify the name of the executable file. Specify this executable file name in the import definition. For execution of the UOC routine, make sure that the path name of the directory containing the executable file has been set in the `PATH` environment variable.

For the UOC routine object name, specify the name of the object file created by compilation.

The libraries under `/opt/hirdbds/lib` are provided by the target Datareplicator. Make sure that these libraries are specified.

Note 2:

The executable file specified in the linkage command cannot have a name that begins with `hds` or that is prohibited by programs already installed in the system.

Note 3:

The `-L` option line contains the libraries that are used for executing SQL statements from within the UOC routine. For the underlined value, specify the directory in which the target HiRDB is installed.

(d) In Linux

- Compilation of a UOC routine in an environment for the 32-bit or IPF version

```
gcc -c -I /HiRDB/include -I /opt/hirdbds/include UOC-source-file-name
```

- Compilation of a UOC routine in an environment for the EM64T version

```
gcc -c -m32 -I /HiRDB/include -I /opt/hirdbds/include UOC-source-file-name
```

`-c:`

Object creation option

`-I /HiRDB/include:`

For the underlined part, specify the target HiRDB installation directory.

`-I /opt/hirdbds/include:`

Directory that stores the header files provided by Datareplicator (always `/opt/hirdbds/include`). You can specify multiple directories if the UOC routine requires other header files.

UOC-source-file-name:

Name for the source file that will be obtained from the SQL preprocessor in a format supported by the C compiler

For details about the syntax of the compile command, see the applicable OS reference manual.

- Linkage of a UOC routine in an environment for the 32-bit or IPF version

```
gcc -o UOC-routine-name -u main UOC-routine-object-name
/opt/hirdbds/lib/libhdsuif.a /opt/hirdbds/lib/libhdsulb.so
/opt/hirdbds/lib/libhdscom.so /opt/hirdbds/lib/libhdspsc.so
-lpthread#1 -ldl
-L/HiRDB/client/lib -lzcltk64#2 -Wl,-R/HiRDB/client/lib
```

#1

In the 32-bit edition, -lpthread is not required.

#2

In the 32-bit edition, replace -lzcltk64 with -lzclt.

- Linkage of a UOC routine in an environment for the EM64T version

```
gcc -o UOC-routine-name -m32 -u main UOC-routine-object-name
/opt/hirdbds/lib/libhdsuif.a /opt/hirdbds/lib/libhdsulb.so
/opt/hirdbds/lib/libhdscom.so /opt/hirdbds/lib/libhdspsc.so
-lpthread -ldl
-L/HiRDB/client/lib -lzclt -Wl,-R/HiRDB/client/lib
```

Note 1:

You can omit the -R option. When you omit the -R option, you must add *HiRDB-installation-directory/lib* to LD_LIBRARY_PATH.

Note 2:

-l specifies the target HiRDB's linkage library name.

Note 3:

The -L option line contains the libraries that are used for executing SQL statements from within the UOC routine. For the underlined part, specify the target HiRDB installation directory.

8.1.3 Creating an import information editing UOC routine (Windows)

This section explains the procedure for creating an import information editing UOC routine with Windows Datareplicator. A UOC routine for Windows Datareplicator is created as a DLL file (*xxx.dll*).

(1) How to create

In the Windows edition, you can create an import information editing UOC routine in the following development environments:

- Microsoft Visual C++ Version 5.0 or later (preprocessor: 32-bit)
- Platform SDK February 2003 or later (preprocessor: IPF)
- Visual Studio 2005 or later (preprocessor: EM64T)

To create a DLL file:

1. Select a table that will use the import information editing UOC routine.
2. Select functions for your import information editing UOC routine; create these functions in the C language that is used to create application programs for Windows HiRDB. To issue SQL statements from the UOC routine, use the HiRDB preprocessor to expand the SQL statements into the applicable language. Datareplicator's UOC header file is stored in `\include` under the Datareplicator installation directory.
3. Compile and link the created functions to create a DLL file. To call the UOC routine from Datareplicator, you must create an interface function with the `__cdecl` calling convention, and then export it.
4. Specify in the import definition as an absolute pathname the name of the import information editing UOC routine.

(2) Compiling and linking

This subsection explains how to compile and link a UOC routine in each development environment.

(a) Microsoft Visual C++ Version 5.0

If you use Microsoft Visual C++ Version 5.0 to compile and link your UOC routine, select **Settings** from the **Project** menu to specify options.

The following table lists and describes the items to be specified in **Project Settings** or **Settings**.

Table 8-1: Items to be specified in Project Settings or Settings in Microsoft Visual C++ Version 5.0

Item	Category	Category setting	Setting
Compiler	Code Generation	Struct Member Alignment	8 bytes
		Runtime Library	Multi-threaded DLL
		Calling Convention	_cdecl *
	Preprocessor	Include file path	<i>Datareplicator-installation-directory</i> \include
Linker	Input	Object/runtime module	CLTDLL.LIB (specify if SQL statements will be issued from the UOC routine)
		Additional library path	<i>HiRDB-installation-directory</i> \CLIENT\LIB (specify if SQL statements will be issued from the UOC routine)

If an unresolved link error occurs, specify all libraries required during linking.

(b) Platform SDK February 2003

- UOC routine compilation

```
cl.exe /c /D_MT /D_DLL /MD /D"WINVER=0x0400" /D"_WIN32_WINNT=0x0333"
/D"_WINNT" /D"NDEBUG" /D"_WINDOWS" /D"WIN32" /D"WIN64" /nologo
/I Datareplicator-installation-directory\include
UOC-source-file-name[UOC-source-file-name]...
```

- UOC routine linkage

```
link.exe /subsystem:console /incremental:no /machine:IA64 /out:
DLL-name-of-import-information-editing-UOC-routine
/DLL /DEF:definition-file UOC-routine-object-name
```

If SQL statements will be issued from within the UOC routine, the library specification *HiRDB-installation-directory*\CLIENT\LIB\CLTDLL.LIB is also required.

(c) Visual Studio 2005

If you use Visual Studio 2005 to compile and link your UOC routine, select **Settings** from the **Project** menu to specify options.

The following table lists and describes the items to be specified.

Table 8-2: Items to be specified in Project Settings or Settings in Visual Studio 2005

Item	Category	Category setting	Setting
Platform	--	--	Win32
Configuration Properties	General	Common Language Runtime Support	No Common Language Runtime Support
Compiler	Code Generation	Struct Member Alignment	8 bytes
		Runtime Library	Multi-threaded DLL
	Advanced	Calling Convention	_cdecl *
	General	Additional Include Directories	<i>Datareplicator-installation-directory\include</i>
Linker	General	Additional Library Directories	<i>HiRDB-installation-directory\CLIENT\LIB</i> (specify if SQL statements will be issued from the UOC routine)
	Input	Additional Dependencies	CLTDLL.LIB (specify if SQL statements will be issued from the UOC routine)

Legend:

--: Not applicable

If an unresolved link error occurs, specify all libraries required during linking.

(3) Notes

- Check that the created UOC routine does not use the same memory mapped file that is used by the target Datareplicator or by any other system.
- If the UOC routine's processing results in an error, the name of the DLL file is displayed in the error log file as the program name.

- If you will be executing transaction control functions from the UOC routine to import data using the two-phase commit method, link the following import libraries provided by Datareplicator:

Datareplicator-installation-directory\lib\hdsuif.lib

- If SQL statements are issued from the UOC routine, specify the HiRDB-provided include files and import libraries during compilation and linking. Also, select the appropriate import libraries according to the transaction control method.

- One-phase commit method

cltdll.lib

- Two-phase commit method

pdcltx32.lib

8.1.4 Syntax for the functions used with an import information editing UOC routine

You use C language to program an import information editing UOC routine. The target Datareplicator provides functions for use in creating your import information editing UOC routine. The following table lists the names of functions that can be used by an import information editing UOC routine and provides an overview of the functions.

Table 8-3: Overview of the functions provided for creating an import information editing UOC routine

Function name	Function
hds_ubegin() (Editing start instruction)	Issues an instruction to start editing of import information. The user is responsible for establishing connection with the target HiRDB or for initializing the environment.
hds_uedit() (Editing and processing instruction)	Issues an instruction to edit, process, or execute synchronization point processing on the import information. The user is responsible for editing or processing the import information, issuing SQL statements, and outputting the information to a general file.
hds_uend() (Editing termination instruction)	Issues an instruction to terminate the editing of input information. The user is responsible for disconnecting from the target HiRDB or for executing termination processing on the environment.

(1) *hds_ubegin()* (editing start instruction function)

(a) Function format

```
#include<hds_ucommon.h>
#include<hds_ureflect.h>
int hds_ubegin(UINTERFACE_BLK *interface-block-name, long *status);
```

(b) Explanation of the parameters

UINTERFACE_BLK **interface-block-name*

To inherit information to the `hds_uedit()` or `hds_uend()` function, specify the name of the area used to store the information. The caller allocates UINTERFACE_BLK. The following table shows the contents of the interface block.

Table 8-4: Contents of the interface block (UINTERFACE_BLK)

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
auth_id	8	char	Authorization identifier	Caller	Connection authorization identifier specified in the import system definition
password	30	char	Password	Caller	Password specified in the import system definition
exsysid	2	char	Source Datareplicator identifier	Extractor	Source Datareplicator identifier specified in the source Datareplicator's extraction system definition, expressed as a string of lowercase hexadecimal characters. During data linkage with XDM/DS, 00 is set.
repid1	2	char	Identifier 1	Caller	Data linkage identifier specified in the import system definition
repid2	2	char	Identifier 2	Caller	Target Datareplicator identifier specified in the import system definition
*inherinf1	4	char *	Inherited information 1	Caller	Address of the area to be inherited to the <code>hds_uedit()</code> and <code>hds_uend()</code> functions
inherinf2	4	long	Inherited information 2	Caller	Information to be inherited to the <code>hds_uedit()</code> and <code>hds_uend()</code> functions
stopinf	4	long	Termination mode	Caller	UOC routine termination mode (this area is effective only during the <code>hds_uend()</code> function call): HDS_FLG_NORMAL Normal termination HDS_FLG_FORCE Forced termination

Notes:

1. Use the header file provided for `UINTERFACE_BLK` by the target Datareplicator. For details about the header files, see Table 8-14 *Header files for an import information editing UOC routine*.
2. End users are not to modify the area that is defined by the caller.

`long *status`

Specify a status value to be displayed in the event of an error in the editing start instruction function. This status value will be displayed in the message that is output by the caller Datareplicator.

(c) Return value

Set the return value when control is returned to the caller. The following table shows the permissible return values.

Table 8-5: Return values from `hds_ubegin()`

Code	Status	Mnemonic	Caller's action
0	Normal termination	HDS_RET_NORM	Resumes processing.
4	Minor error detected (resumable level)	HDS_RET_WARN	Outputs a message and resumes processing.
Other	Major error detected (unresumable level)	HDS_RET_ERR	Outputs a message, cancels processing, and terminates the process.

(d) Notes on using the source Datareplicator

If you have specified a transmission-suppressed original receiver identifier in the `ndndidxxx` operand (`xxx`: integer in the range 001 to 256) in the source Datareplicator's transmission environment definition, specify in the `$PDCLTAPNAME` HiRDB environment variable immediately before HiRDB's `CONNECT` processing a character string of up to 20 bytes beginning with `hds_sqlc` and ending with the source Datareplicator identifier.

The character string specified as the source Datareplicator identifier is the same as the one that is specified in the interface block (2-byte hexadecimal character string).

(2) *hds_uedit()* (editing and processing instruction function)

(a) Function format

```
#include<hds_ucommon.h>
#include<hds_ureflect.h>
int hds_uedit(UINTERFACE_BLK *interface-block-name ,
             UREFLECT_BLK *import-control-block-name ,
             UDATA_BLK *key-information-block-name ,
             UDATA_BLK *import-data-block-name ,
             long *status) ;
```

(b) Explanation of the parameters

UINTERFACE_BLK **interface-block-name*

To inherit information from the `hds_begin()` function, specify the name of the area used to store the information. For details about the interface block names, see Table 8-4 *Contents of the interface block (UINTERFACE_BLK)*.

UREFLECT_BLK **import-control-block-name*

Specify the name of the area used to store common import information. The caller allocates the import control block. The following table shows the contents of the import control block.

Table 8-6: Contents of the import control block (UREFLECT_BLK)

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
updkind	8	char	Update type	Caller	Type of update processing on the source database (as an 8-byte character string, as shown below; if the type is shorter than 8 bytes, spaces are added): INSERT Indicates INSERT. UPDATE Indicates UPDATE. DELETE Indicates DELETE. PURGE Indicates PURGE. COMMIT Indicates COMMIT [#] . [#] : When this value is stored, the values following the authorization identifier serve no purpose.
auth_id	30	char	Authorization identifier	Caller	Authorization identifier used to update the source database
tbl_id	30	char	Table identifier	Caller	Table identifier used to update the source database
upd_date	4	char	Update date	Caller	Date the source database was updated (date the update information was output to the journal). The date is displayed in unsigned packed decimal format (YYYYMMDD).

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
upd_time	4	char	Update time	Caller	Time the source database was updated (time the update information was output to the journal). The time is displayed in unsigned packed decimal format (<i>HHMMSSTT</i> , where <i>TT</i> is 1/100 sec.).
uapname	8	char	Source UAP name	Caller	Name of the user application program used to update the source database
extsysid	4	char	Source database's system ID	Caller	When the source database is a HiRDB: Source database's HiRDB identifier When the source database is a mainframe (XDM/DS, PDM2 E2, or RDB1 E2): Source database's subsystem ID
infflag1	1	unsigned char	Reserved area1	N/A	Reserved
infflag2	1	unsigned char	Reserved area2	N/A	Reserved
infflag3	1	unsigned char	Reserved area3	N/A	Reserved
reserve1	1	char	Reserved area4	N/A	Reserved
infflag4	4	long	Reserved area5	N/A	Reserved

Notes:

1. Use the header file provided for UREFLECT_BLK by the target Datareplicator. For details about the header files, see Table 8-14 *Header files for an import information editing UOC routine*.
2. End users are not to modify the area that is defined by the caller.

UDATA_BLK **key-information-block-name*

UDATA_BLK **import-data-block-name*

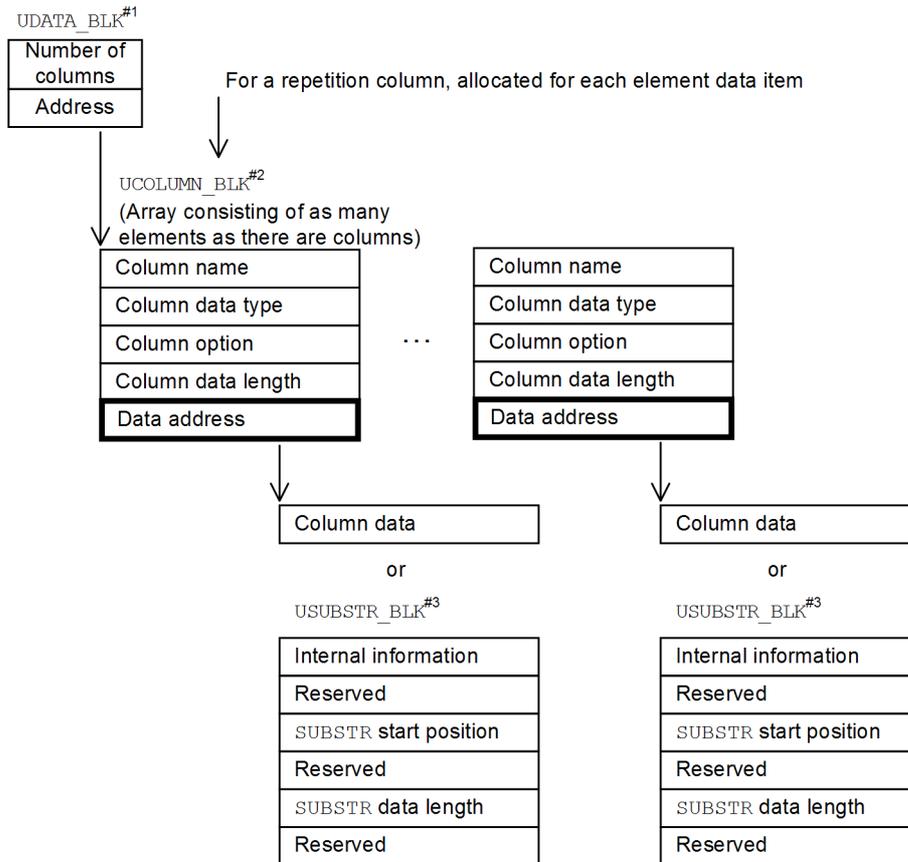
Specify the name of the area used to store the import information. This structure is used with the key information block in the third parameter and the import data

block in the fourth parameter of the `hds_uedit()` function. In the key information block, the mapping key information is stored in `UDATA_BLK`; in the import data block, the import data information is stored in `UDATA_BLK`.

For the key information block, 0 is set as the number of columns in the case of `INSERT`, `PURGE`, or `COMMIT`. If the key value is changed, the key value before the update processing is set in the key information block. The key value after update processing or during `INSERT` processing is set in the import data block.

In the case of `DELETE`, `PURGE`, or `COMMIT`, 0 is set as the number of columns in the import data block. The key information and import data information consist of the structure pointed to by `UDATA_BLKs` (this is referred to as `UCOLUMN_BLK`) and scalar data. If the number of columns is 0 in `UDATA_BLK`, do not reference `UCOLUMN_BLK`. The following figure shows `UDATA_BLK` and related structures.

Figure 8-4: `UDATA_BLK` and related structures



#1

For the contents of UDATA_BLK, see Table 8-7 *Contents of UDATA_BLK*.

#2

For the contents of UCOLUMN_BLK, see Table 8-8 *Contents of UCOLUMN_BLK*.

#3

This data structure applies to the BLOB or BINARY type with backward deletion updating. For the contents of USUBSTR_BLK, see Table 8-9 *Contents of USUBSTR_BLK*.

Table 8-7: Contents of UDATA_BLK

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
colnum	4	long	Number of columns	Caller	Number of columns for the mapping key or import data
*colinfptr	4	UCOLUMN_BLK *	UCOLUMN_BLK * address	Caller	Address of UCOLUMN_BLK

Note:

Use the header file provided for UDATA_BLK by Datareplicator. For details about the header files, see Table 8-14 *Header files for an import information editing UOC routine*.

Table 8-8: Contents of UCOLUMN_BLK

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
colname	30	char	Column name	Caller	Column name
coltype	1	unsigned char	Column data type	Caller	Column's data type. For details about the settings, see Table 8-10 <i>Column data type mnemonics</i> .

8. User Own Coding Routines

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
coloption	1	unsigned char	Column option	Caller	<p>Column options, as shown below:</p> <p>HDS_COL_NOOPT (0x00): No options</p> <p>HDS_COL_NULL (0x01) Null data</p> <p>HDS_COL_MAPK (0x02): Mapping key: HDS_COL_MAPK is set only in the import data block. It is not set in the key information block.</p> <p>HDS_COL_ARRAY (0x04): Array column</p> <p>HDS_COL_REPET (0x08): Repetition column</p> <p>HDS_COL_ELMNL (0x10): Repetition column's element value is null.</p> <p>HDS_COL_REPTCOL (0x20): Specification of repetition structure</p> <p>HDS_COL_BLOBSUB (0x40): Concatenation operation data</p> <p>HDS_COL_SUBSTR (0x80): SUBSTR operation backward deletion update log</p>

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
elementnum	4	union	Repetition column's valid element number or number of abstract data types	Caller	<p>For a repetition column: The valid element number is set in the first two bytes (short). If an asterisk (*) is specified, -2 is set. The number of element data items in the corresponding column is set in the last two bytes (short).</p> <p>For an abstract data type column: The number of data types is set, including the inheritance relationship. If there is no inheritance, 1 is set.</p> <p>For an array column: The element number of the source array column is set in the 2 leading bytes. 0 is set in the 2 trailing bytes.</p>
mltcolkind	1	unsigned char	Repetition column update type	Caller	<p>Update type for a repetition column, as shown below (this information is applicable only when the update type is UPDATE):</p> <p>HDS_COL_ADDV (0x01): ADD HDS_COL_DELV (0x02): DELETE HDS_COL_SETV (0x04): SET</p>
adtfunc	1	char	Abstract data type import method	Caller	Always 0x01.
charset	1	char	Character set type	Caller	<p>Character set type of the column data that is to be passed to the UOC routine</p> <p>When no character set is specified: X'00' (HDS_CSET_DEFAULT)</p> <p>When a character set (EBCDIK) is specified: X'01' (HDS_CSET_EBCDIK)</p>

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
reserve1	2	char	Reserved area	N/A	Reserved
collen	4	long	Column data length	Caller	Column's data length. If the column option is null data, 0 is set.
dataptr	4	union	Data address	Caller	Address of the column data. If the column option is null data, the null pointer is set. For details about the settings, see Table 8-11 <i>Data address mnemonics</i> .

Note:

Use the header file provided for UCOLUMN_BLK by the target Datareplicator. For details about the header files, see Table 8-14 *Header files for an import information editing UOC routine*.

Table 8-9: Contents of USUBSTR_BLK

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
reserve1	4	int	Internal information 1	Caller	Internal information
data_len	4	int	Internal information 2	Caller	Internal information
reserve2	4	int	Reserved area 1	Caller	Reserved
substr_spos	4	unsigned int	SUBSTR start position	Caller	Start position of the data column to be obtained when the SUBSTR scalar function is used
reserve3	4	int	Reserved area 2	Caller	Reserved
substr_len	4	unsigned int	SUBSTR data length	Caller	Length of the data column to be obtained when the SUBSTR scalar function is used
reserve4	16	char	Reserved area 3	Caller	Reserved

Table 8-10 *Column data type mnemonics* shows the column data type mnemonics, and Table 8-11 *Data address mnemonics* shows the data address mnemonics.

Table 8-10: Column data type mnemonics

Column data type	Mnemonic	Data code
INTEGER	HDS_T_INT	(0xF1)
SMALLINT	HDS_T_SINT	(0xF5)
LARGE DECIMAL	HDS_T_DEC	(0xE5)
FLOAT	HDS_T_FLT	(0xE1)
DOUBLE PRECISION	HDS_T_DBL	(0xE1)
SMALL FLOAT	HDS_T_SFLT	(0xE3)
REAL	HDS_T_REAL	(0xE3)
CHARACTER	HDS_T_CHAR	(0xC5)
VARCHAR	HDS_T_VCHAR	(0xC1)
NCHAR	HDS_T_NCHAR	(0xB5)
NVARCHAR	HDS_T_NVCHAR	(0xB1)
MCHAR	HDS_T_MCHAR	(0xA5)
MVCHAR	HDS_T_MVCHAR	(0xA1)
DATE	HDS_T_DATE	(0x71)
TIME	HDS_T_TIME	(0x79)
TIMESTAMP	HDS_T_TIMESTAMP	(0x7D)
INTERVAL YEAR TO DAY	HDS_T_YTD	(0x65)
INTERVAL HOUR TO SECOND	HDS_T_HTS	(0x6F)
BLOB	HDS_T_BLOB	(0x93)
BINARY	HDS_T_BINARY	(0x91)
UNPACK	HDS_T_UNPACK	(0xFF)

Table 8-11: Data address mnemonics

Data type	Mnemonic	Address type
INTEGER	HDS_A_INT	long *
SMALLINT	HDS_A_SINT	short *

Data type	Mnemonic	Address type
LARGE DECIMAL	HDS_A_DEC	char *
FLOAT	HDS_A_FLT	double *
DOUBLE PRECISION	HDS_A_DBL	double *
SMALL FLOAT	HDS_A_SFLT	float *
REAL	HDS_A_REAL	float *
CHARACTER	HDS_A_CHAR	char *
VARCHAR	HDS_A_VCHAR	char *
NCHAR	HDS_A_NCHAR	char*
NVARCHAR	HDS_A_NVCHAR	char *
MCHAR	HDS_A_MCHAR	char *
MVCHAR	HDS_A_MVCHAR	char *
DATE	HDS_A_DATE	char *
TIME	HDS_A_TIME	char *
INTERVAL YEAR TO DAY	HDS_A_YTD	char *
INTERVAL HOUR TO SECOND	HDS_A_HTS	char *
BLOB	HDS_A_BLOB	char *
UNPACK	HDS_A_UNPACK	char *
TIMESTAMP	HDS_A_TIMESTAMP	char *
BINARY	HDS_A_BINARY	char *
SUBSTR operation backward deletion update log	HDS_A_SUBSTR	USUBSTR_BLK *

long **status*

Specify a status value to be displayed in the event of an error in the editing and processing instruction function. This status value will be displayed in the message that is output by the caller interface process after control is returned from the function.

(c) Return value

Set the return value when control is returned to the caller. The following table shows the permissible return values.

Table 8-12: Return values from hds_uedit()

Code	Status	Mnemonic	Caller's action
0	Normal termination	HDS_RET_NORM	Resumes processing.
4	Minor error detected (resumable level)	HDS_RET_WARN	Outputs a message and resumes processing.
Other	Major error detected (unresumable level)	HDS_RET_ERR	Outputs a message, cancels processing, and terminates the process.

(3) hds_uend() (editing termination instruction function)**(a) Function format**

```
#include<hds_ucommon.h>
#include<hds_ureflect.h>
int hds_uend(UINTERFACE_BLK *interface-block-name ,
            long *status);
```

(b) Explanation of the parameters

UINTERFACE_BLK *interface-block-name

To inherit information from the hds_begin() or hds_edit() function, specify the name of the area used to store the information. For details about the interface block names, see Table 8-4 *Contents of the interface block (UINTERFACE_BLK)*.

If you have allocated the area dynamically with the hds_ubegin() function, you must release it within the hds_uend() function. Close any file that was opened. Once control is returned from the hds_uend() function, release the area for the structure without doing anything to the user-specified item in UINTERFACE_BLK.

If the termination mode of the UOC routine accessing the database is forced termination, execute ROLLBACK on the DBMS, and then execute DISCONNECT.

long *status

Specify a status value to be displayed in the event of an error in the editing termination instruction function. This status value will be displayed in the message that is output by the caller interface process after control is returned from the function.

(c) Return value

Set the return value when control is returned to the caller. The following table shows the permissible return values.

Table 8-13: Return values from hds_uend()

Code	Status	Mnemonic	Caller's action
0	Normal termination	HDS_RET_NORM	Resumes processing.
4	Minor error detected (resumable level)	HDS_RET_WARN	Outputs a message and terminates the process normally.
Other	Major error detected (unresumable level)	HDS_RET_ERR	Outputs a message and terminates the process normally.

(4) Header files used with an import information editing UOC routine

A target Datareplicator provides header files for the structures and mnemonics that are used as the interface for the `hds_ubegin()`, `hds_uedit()`, and `hds_uend()` functions.

The header files used with an import information editing UOC routine are stored in the following directory:

UNIX Datareplicator: `/opt/hirdbds/include/`

Windows Datareplicator: `Datareplicator-installation-directory\include\`

The following table lists and describes the header files for an import information editing UOC routine.

Table 8-14: Header files for an import information editing UOC routine

Filename	Description	Members associated with the import information editing UOC
<code>hds_ucommon.h</code>	Structures and mnemonic codes common to import information editing UOC routines	<ul style="list-style-type: none"> • <code>UINTERFACE_BLK</code> • Mnemonics for the return value
<code>hds_ureflect.h</code>	Structures and mnemonics for an import information editing UOC routine	<ul style="list-style-type: none"> • <code>UREFLECT_BLK</code> • Mnemonics for <code>UREFLECT_BLK</code> • <code>UDATA_BLK</code> • <code>UCOLUMN_BLK</code> • Mnemonics for the data type • Mnemonics for the data address

(5) Rules

(a) Files

You can manipulate only user-specific files (that are open) within the functions of an import information editing UOC routine. The following are the file-related rules:

- Do not use the standard input, standard output, or standard error output.

If you have specified `FALSE` in the target Datareplicator's `HDS_MST_STDCLOSE` environment variable, you can use the standard input, standard output, and standard error output.

- Do not write to or read from any file other than those opened by the import information editing UOC routine.
- Do not manipulate any file that belongs to the target Datareplicator.

(b) Relationship with definitions

- A `control_trigger` instruction is not issued to the UOC routine. Therefore, when the UOC routine is used, the `control_trigger` parameter in the target environment definition is ignored.
- Because Datareplicator does not issue the `disconnect` instruction to UOC routines, the `discintvl` operand specification in the import system definition does not apply to import processing executed by a UOC routine.
- Because Datareplicator does not check the table used within a UOC routine, the `tblcheck` operand specification in the import environment definition does not apply to import processing executed by a UOC routine.
- To execute import processing with a UOC routine, specify the name of the user-created UOC routine in the import definition. During import processing, the UOC routine whose name is specified is started automatically.
- If you select all UOC routines to execute import processing in the import definition, manipulations on a time-ordered information table will not take place automatically.

(c) Specifications in the SQL descriptor area

To use the SQL descriptor area to update the target database, you must create the UOC routine by setting the necessary information from the UOC interface table into the *SQL descriptor area*. For the UOC routine to directly reference column data in the UOC interface table, it might be necessary to code in C language, depending on the data type. The following figure shows the relationships among the format of data indicated by the column data address in the UOC interface table, the length of real data, and the representation in C language.

Table 8-15: Relationships among the UOC routine's column data type, data length, and representation in C language

UOC interface table		Size of data area (bytes)	Representation in C language
Mnemonic	Length of column data		
HDS_T_INT	4	4	long

8. User Own Coding Routines

UOC interface table		Size of data area (bytes)	Representation in C language
Mnemonic	Length of column data		
HDS_T_SINT	2	2	short
HDS_T_DEC	Leading 2 bytes: precision Trailing 2 bytes: scaling	$Precision \div 2 + 1$	char <i>data-area</i> [<i>real-data-length</i>]
HDS_T_FLT	8	8	double
HDS_T_DBL			
HDS_T_SFLT	4	4	float
HDS_T_REAL			
HDS_T_CHAR	Length of character string (bytes)	<i>UOC routine's column data length</i>	char <i>data-area</i> [<i>real-data-length</i>]
HDS_T_MCHAR			
HDS_T_VCHAR	Length of character string in the data section (bytes)	<i>UOC routine's column data length + 2</i>	struct{ unsigned short <i>real-data-length</i> char <i>data-area</i> [<i>real-data-length</i>] }
HDS_T_NCHAR	Length of character string (characters)	<i>UOC routine's column data length x 2</i>	char <i>data-area</i> [<i>real-data-length</i>]
HDS_T_NVCHAR	Length of character string in the data section (characters)	<i>UOC routine's column data length x 2 + 2</i>	struct{ unsigned short <i>real-data-length</i> char <i>data-area</i> [<i>real-data-length</i>] }
HDS_T_DATE	4	4	char <i>data-area</i> [<i>real-data-length</i>]
HDS_T_TIME	3	3	char <i>data-area</i> [<i>real-data-length</i>]
HDS_T_YTD	Leading 2 bytes: precision (8) Trailing 2 bytes: scaling (0)	5	char <i>data-area</i> [<i>real-data-length</i>]
HDS_T_HTS	Leading 2 bytes: precision (6) Trailing 2 bytes: scaling (0)	4	char <i>data-area</i> [<i>real-data-length</i>]

UOC interface table		Size of data area (bytes)	Representation in C language
Mnemonic	Length of column data		
HDS_T_BLOB (when the column option is not 0x80)	Length of data in the data section (bytes)	<i>UOC routine's column data length</i> + 8	struct{ long <i>reserved-area</i> unsigned long <i>real-data-length</i> char <i>data-area</i> [<i>real-data-length</i>] }
HDS_T_BLOB (when the column option is 0x80)	32	<i>UOC routine's column data length</i> + 8	Table 8-9 Contents of <i>USUBSTR_BLK</i>
HDS_T_UNPACK	Length of UNPACK data (bytes)	<i>Length of integer part</i> + <i>length of fraction part</i>	char <i>data-area</i> [<i>real-data-length</i>]
HDS_T_TIMESTAMP	7 + ↑ <i>p</i> /2 ↑ (bytes) <i>p</i> : Number of decimal places	<i>UOC routine's data column data length</i>	char <i>data-area</i> [<i>real-data-length</i>]
HDS_T_BINARY (when the column option is not 0x80)	Length of data in the data section (bytes)	<i>UOC routine's data column data length</i> + 4	struct{ long <i>real-data-length</i> char <i>data-area</i> [<i>real-data-length</i>] }#
HDS_T_BINARY (when the column option is 0x80)	32	<i>UOC routine's column data length</i> + 8	Table 8-9 Contents of <i>USUBSTR_BLK</i>

Notes:

In the case of data in a repetition column, the data for one element is stored in one UCOLUMN_BLK.

#

The transfer method is the same for concatenation operations.

(d) Data types

- If the source database is a mainframe database and data conversion is required, the data obtained after data conversion is passed to the UOC routine. For details about data conversion, see 4.3.4 *Designing for the supported data types*.
- DATE-type and TIME-type data are stored in unsigned packed format. When you create a UAP that uses the SQL descriptor area, you can specify the address of the data storage area directly in the SQL descriptor area. When you create a UAP that

does not use the SQL descriptor area, you must convert DATE-type and TIME-type data to the corresponding date and time character string data, and then specify them in embedded variables.

- SMALLFLOAT-type data is passed as FLOAT-type data to avoid overflow that might occur due to a difference in representation ranges.
- If an extracted column's type is UNPACK, Datareplicator converts it to DECIMAL-type data, and then passes it to the UOC routine in DECIMAL data format. If Datareplicator cannot convert the UNPACK data to DECIMAL, it passes the UNPACK data to the UOC routine. For details about the mnemonic for the UNPACK-type that is passed to the UOC routine, see Table 8-10 *Column data type mnemonics* or Table 8-11 *Data address mnemonics*.

8.1.5 Handling of abstract data types by an import information editing UOC routine

If abstract data types are passed to an import information editing UOC routine, the input data for constructor functions of the abstract data type is passed as attribute data of the abstract data type.

Because attribute data of the abstract data type contains information that is specific to the abstract data type, an import information editing UOC routine must import it to the constructor functions of the abstract data type as is (update data of the abstract data type cannot be edited or processed before it is imported).

(1) *Abstract data type mnemonics and the structure of column data*

This subsection explains the abstract data type mnemonics and the structure of column data.

(a) **Abstract data type mnemonics**

Table 8-16 *Column data type mnemonics (UCOLUMN_BLK.coltype)* and Table 8-17 *Data address mnemonics (UCOLUMN_BLK.dataptr)* show the abstract data type mnemonics. For details about the non-abstract data type mnemonics, see Table 8-10 *Column data type mnemonics* and Table 8-11 *Data address mnemonics*.

Table 8-16: Column data type mnemonics (UCOLUMN_BLK.coltype)
(UCOLUMN_BLK.coltype)

Column data type	Mnemonic	Data code
ADT (abstract data type)	HDS_T_ADT	(0x83)

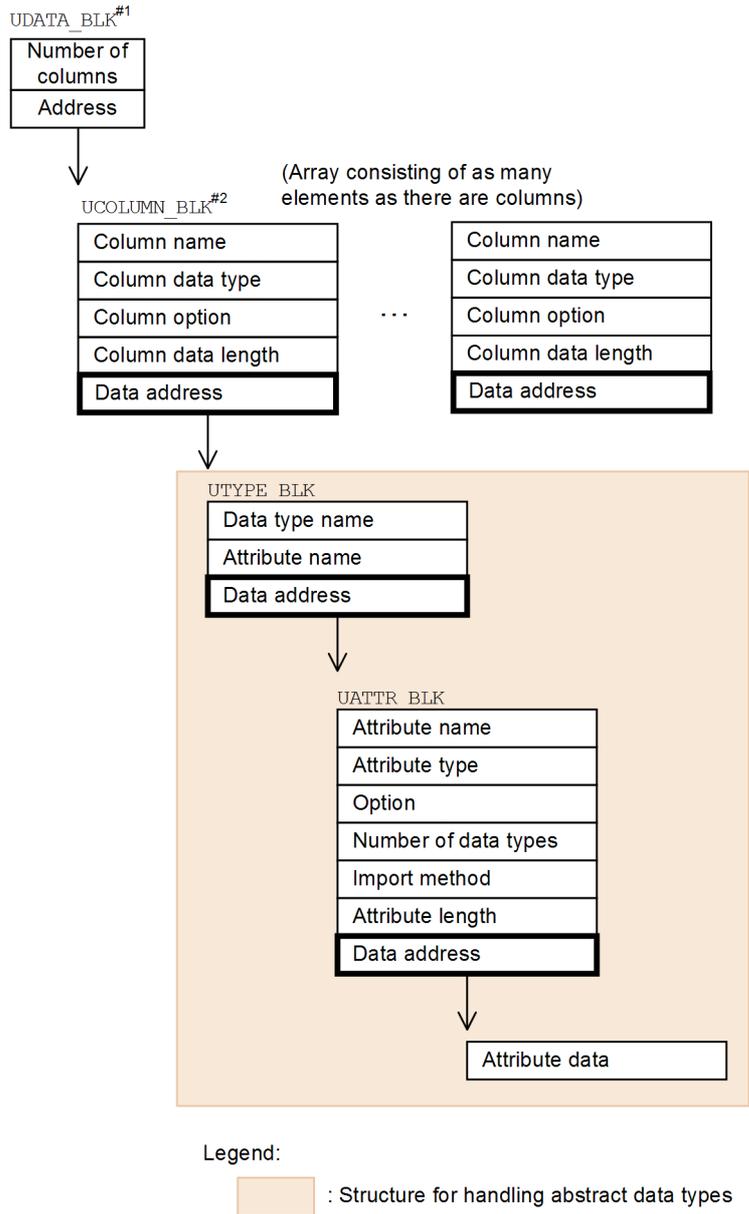
Table 8-17: Data address mnemonics (UCOLUMN_BLK.dataptr)

Data type	Mnemonic	Address type
ADT (abstract data type)	HDS_A_ADT	UTYPE_BLK *

(b) Structure of abstract data-type column data

The figure below shows the structure of abstract data-type column data. For details about the structure of non-abstract data-type column data, see Figure 8-4 *UDATA_BLK and related structures*.

Figure 8-5: Structure of abstract data-type column data



#1

For the contents of UDATA_BLK, see Table 8-7 Contents of UDATA_BLK.

#2

For the contents of UCOLUMN_BLK, see Table 8-8 *Contents of UCOLUMN_BLK*.

The following table shows the contents of UTYPE_BLK (abstract data type information).

Table 8-18: Contents of UTYPE_BLK (abstract data type information)

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
schema_name	31	char	Authorization identifier	Caller	Authorization identifier for the abstract data type (including the termination symbol)
type_name	31	char	Abstract data type name	Caller	Type name of the abstract data type (including the termination symbol)
reserve1	2	char	Reserved area 1	Caller	Reserved
def_attrnum	4	int	Number of defined attributes	Caller	Number of attributes defined for the abstract data type
set_attrnum	4	int	Number of attribute data items	Caller	Number of attribute data items stored in the data type
data_addr	4	UATTR_BLK *	UATTR_BLK * address	Caller	Address of the attribute management table (UATTR_BLK)
reserve2	8	char	Reserved area 2	Caller	Reserved

The following table shows the contents of UATTR_BLK (attribute data information).

Table 8-19: Contents of UATTR_BLK (attribute data information)

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
attr_name	31	char	Attribute name	Caller	Attribute name (including the termination symbol)
attr_type	1	unsignedchar	Data type of the attribute	Caller	Data type of the attribute For details about the data type mnemonics, see Table 8-10 <i>Column data type mnemonics</i> .

8. User Own Coding Routines

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
null_flag	1	unsignedchar	NULL flag for the attribute	Caller	NULL flag for the attribute <ul style="list-style-type: none"> 0x01: NULL value 0x00: Non-NULL value
adtfunc	1	unsignedchar	Import method for the abstract data type	Caller	Import method for the abstract data type <ul style="list-style-type: none"> 0x01: Constructor function import method 0x00: The data type of the attribute is not the abstract data type (0x83)
reserve1	3	char	Reserved area 1	Caller	Reserved
attr_len	4	int	Attribute data length	Caller	Attribute data length The value is 0 in the following cases: <ul style="list-style-type: none"> The data type of the attribute is the abstract data type The NULL flag of the attribute is the NULL value
adtelnum	4	int	Abstract data type inheritance count	Caller	Number of data types with an inheritance relationship when the data type of the attribute is the abstract data type <ul style="list-style-type: none"> 1: No inheritance 0: The data type of the attribute is not the abstract data type (0x83)
data_addr	4	UNION *	Attribute data address	Caller	Attribute data address (input data for constructor function of the abstract data type that is entered at the source) <ul style="list-style-type: none"> UTYPE_BLK address: The data type of the attribute is the abstract data type NULL: The NULL flag of the attribute is the NULL value For details about the data address mnemonics, see Table 8-11 <i>Data address mnemonics</i> .
reserve2	8	char	Reserved area 2	Caller	Reserved

(c) Settings for UTYPE_BLK and UATTR_BLK of each abstract data type

Table 8-20 *Settings for UTYPE_BLK for each abstract data type* shows the settings for UTYPE_BLK for each abstract data type, and Table 8-21 *Settings for UATTR_BLK for each abstract data type* shows the settings for UATTR_BLK for each abstract data type.

Table 8-20: Settings for UTYPE_BLK for each abstract data type

Member name	Abstract data type		
	SGMLTEXT	FREWORD	XML
schema_name	MASTER	MASTER	MASTER
type_name	SGMLTEXT	FREWORD	XML
reserve1	Reserved	Reserved	Reserved
def_attrnum	0	0	0
set_attrnum	1	1	1
data_addr	UATTR_BLK address	UATTR_BLK address	UATTR_BLK address
reserve2	Reserved	Reserved	Reserved

Table 8-21: Settings for UATTR_BLK for each abstract data type

Member name	Abstract data type		
	SGMLTEXT	FREWORD	XML
attr_name	Not specified.	Not specified.	Not specified.
attr_type	HDS_T_BLOB(0x93)#	HDS_T_VCHAR(0xC1)#	HDS_T_BINARY(0x91)#
null_flag	0x00	0x00	0x00
adtfunc	0x00	0x00	0x00
reserve1	Reserved	Reserved	Reserved
attr_len	Actual attribute data length (maximum 2,147,483,647)	Actual attribute data length (maximum 32,000)	Actual attribute data length (maximum 2,147,483,647)
adtelnum	0	0	0
data_addr	Attribute data address	Attribute data address	Attribute data address
reserve2	Reserved	Reserved	Reserved

#

To reference attribute data of the abstract data type that has been passed to an import information editing UOC routine, use a cast in C language in the same manner as for the normal column data types. If you will be casting the data structure indicated by the attribute data address to the attribute data type, see *Table 8-15 Relationships among the UOC routine's column data type, data length, and representation in C language.*

(2) Notes about handling abstract data types

For abstract data types, no value is set in the `colLen` member (column data length) of `UCOLUMN_BLK` that is passed to an import information editing UOC routine. For this reason, when abstract data types are handled, you must not reference the `colLen` member of `UCOLUMN_BLK`.

(3) Sample import information editing UOC routine that handles abstract data types

Configuration of the source and target tables

The figure below shows the structure of the sample table that is subject to extraction processing. The sample target table has the same structure.

Column data types	INTEGER	CHAR (64)	SGMLTEXT	FREWORD	XML
Column names	KEYNO	KEYNAME	COL_SGMLTEXT	COL_FREWORD	COL_XML
	:	:	:	:	:

Example code

The following figure shows an example of coding that executes `INSERT` on the table that contains columns of the `SGMLTEXT`, `FREWORD`, and `XML` types.

Figure 8-6: Example of coding that executes INSERT on the table that contains columns of the abstract data types.

```

/*****
/*   Function = sample import information editing UOC routine that uses
/*   abstract data types
*****/
#include <hds_ucommon.h>
#include <hds_ureflect.h>
#include <stdio.h>

#define SQL_INSERT      "INSERT "

                        :
                        :

/*****
/*   Function = edit and process
*****/
int hds_uedit(
    UINTERFACE_BLK *pstUiBlock,
    UREFLECT_BLK *pstRefBlock,
    UDATA_BLK *pstKeyBlock,
    UDATA_BLK *pstRefDataBlock,
    long *piStatus)
/* for the IPF version, specify int, not long */
{
    EXEC SQL BEGIN DECLARE SECTION;
    int iKeyNo; /* key number (INT) */
    char szKeyName[64]; /* key name (CHAR(64)) */
    SQL TYPE IS BLOB(100000) stSgmltext; /* SGMLTEXT column (#1) */
    SQL TYPE IS BINARY(100000) stXml; /* XML column (#1) */
    SQL TYPE IS VARCHAR(32000) stFreeWord; /* FREEWORD column (#1) */
    /* #1: allocate the required size */
    EXEC SQL END DECLARE SECTION;

    int iRc; /* this function's return code */
    UCOLUMN_BLK *pstColBlk; /* column information */
    UTYPE_BLK *pstTypeBlk; /* abstract data type information */
    UATTR_BLK *pstAttrBlk; /* abstract data type attribute information */

    if ( memcmp( pstRefBlock->updkind, SQL_INSERT, strlen(SQL_INSERT)) == 0 ) {
        /* KEYNO */
        pstColBlk = &(pstRefDataBlock->colinfptr[0]);
        iKeyNo = *(pstColBlk->dataptr.intptr);

        /* KEYNAME */
        pstColBlk = &(pstRefDataBlock->colinfptr[1]);
        memcpy(szKeyName, pstColBlk->dataptr.cptr, 64);

        /* COL_SGMLTEXT */
        /* import the update data passed to the import information editing UOC routine as is */
        pstColBlk = &(pstRefDataBlock->colinfptr[2]);
        pstTypeBlk = (UTYPE_BLK *)pstColBlk->dataptr.adtptr;
        pstAttrBlk = (UATTR_BLK *) (pstTypeBlk->data_addr);
        stSgmltext.stSgmltext_reserved = 0;
        stSgmltext.stSgmltext_length =
            *(unsigned long *) (pstAttrBlk->data_addr.blobptr + sizeof(long));
        memcpy(stSgmltext.stSgmltext_data,
            (pstAttrBlk->data_addr.blobptr + sizeof(long) + sizeof(unsigned long) ),
            stSgmltext.stSgmltext_length);
    }
}

```

```

/*----- Comment Begin (for IPF) -----
for the IPF version, specify int, not long
stSgmltext.stSgmltext_length =
    *(unsigned int*) (pstAttrBlk->data_addr.blobptr + sizeof(int));
memcpy(stSgmltext.stSgmltext_data,
    (pstAttrBlk->data_addr.blobptr + sizeof(int) + sizeof(unsigned int) ),
    stSgmltext.stSgmltext_length);
----- Comment End -----*/

/* COL_XML */
/* import the update data passed to the import */
/* information editing UOC routine as is */
pstColBlk = &(pstRefDataBlock->colinfptr[3]);
pstTypeBlk = (UTYPE_BLK *)pstColBlk->dataptr.adtptr;
pstAttrBlk = (UATTR_BLK *) (pstTypeBlk->data_addr);
stXml.len = *(long *) (pstAttrBlk->data_addr.binaryptr);
memcpy(stXml.str, (pstAttrBlk->data_addr.binaryptr + sizeof(long)), stXml.len);
/*----- Comment Begin (for IPF) -----
for the IPF version, specify int, not long
stXml.len = *(int *) (pstAttrBlk->data_addr.binaryptr);
memcpy(stXml.str, (pstAttrBlk->data_addr.binaryptr + sizeof(int)), stXml.len);
----- Comment End -----*/

/* COL_FREEWORD */
/* import the update data passed to the import */
/* information editing UOC routine as is */
pstColBlk = &(pstRefDataBlock->colinfptr[4]);
pstTypeBlk = (UTYPE_BLK *)pstColBlk->dataptr.adtptr;
pstAttrBlk = (UATTR_BLK *) (pstTypeBlk->data_addr);
stFreeWord.len = *(short *) ((unsigned short *) (pstAttrBlk->data_addr.vcptr));
memcpy(stFreeWord.str,
    (pstAttrBlk->data_addr.vcptr + sizeof(stFreeWord.len)),
    stFreeWord.len);

EXEC SQL
    INSERT INTO ADTTBL( KEYNO, KEYNAME,
        COL_SGMLTEXT,
        COL_XML,
        COL_FREEWORD)
    VALUES ( :iKeyNo, :szKeyName,
        SGMLTEXT ( :stSgmltext AS BLOB(2147483647) ),
        XML ( :stXml AS BINARY(2147483647) ),
        FREEWORD ( :stFreeWord AS VARCHAR(32000) ) );
        .
        .
        .

```

Legend:

 : Manipulation of update data of the abstract data type

8.1.6 Notes on creating an import information editing UOC routine

(1) *Limitation on signals*

Do not conduct signal operations in an import information editing UOC routine's functions.

(2) *Limitations on system calls*

- Do not issue `alarm()` or `pause()` system calls.
- If you generate a subprocess by `fork()` or `exec()`, execute postprocessing on the user side.

(3) *Limitation on shared memory*

Do not use the same shared memory that is used by the target Datareplicator or by any other system.

(4) *Limitation on synchronization points*

If `hds_uedit()` passes `COMMIT` and its return value is 0 or 4, Datareplicator assumes that one synchronization point was completed. If the import information editing UOC routine's process is aborted due to an error, restart takes place at the synchronization point immediately following the previous synchronization point. You must code the processing in the import information editing UOC routine's functions so that synchronization point processing occurs when `COMMIT` is passed and so that no duplicate processing occurs when processing is restarted after an abnormal termination.

(5) *Notes on creating a UOC routine that accesses a database*

When you create a UOC routine that accesses a database, use the `hds_uend()` editing termination function to reference the termination information flag (`stopinf`) in the interface block (`UINTERFACE_BLK`). If the termination information indicates forced termination, create the UOC routine so that it executes `ROLLBACK` on the database system, and then executes `DISCONNECT`. If the UOC routine executes `DISCONNECT` without executing `ROLLBACK`, update processing up to the point of forced termination will be committed by automatic commitment of `DISCONNECT`. The next time Datareplicator is restarted, there will be an inconsistency in the database and the UOC routine will result in an SQL error.

(6) *Relationship with the import environment definition*

- `discintvl` (`disconnect` issuance interval) is not applicable to an import group using the UOC routine.
- If a trigger is defined for the target table and `HiRDB` is accessed within the import information editing interface function, the `control_trigger` parameter in the import environment definition is ignored.

(7) Notes when not collecting an import processing-related audit trail

The notes provided here apply to when you create a new import information editing UOC routine.

If you are using an existing import information editing UOC routine, re-link the executable file of the import information editing UOC routine. In this case, verify that the existing import information editing UOC routine does not violate the notes provided here. For details about the linkage procedure, see *8.1 Import information editing UOC routine*. If you do not link the programs, an audit trail will be collected.

- When you create an import information editing UOC routine, observe the following rules:
 - Execute `CONNECT` to `HiRDB` within the `hds_ubegin` function.
If this rule is violated, the `KFRB03094-W` message will be output.
The subsequent processing depends on the `hirdb_audit_trail` operand value in the import system definition.
 - Do not execute any SQL statement other than `CONNECT` within the `hds_ubegin` function.
If this rule is violated, an audit trail will be collected for any SQL statement specified within the `hds_ubegin` function, except for `CONNECT`. If a transaction has not been completed within the `hds_ubegin` function, an audit trail might be collected for that transaction.
 - Execute `DISCONNECT` to `HiRDB` within the `hds_uend` function.
If this rule is violated, an audit trail will be collected following issuance of `DISCONNECT`.
- If the import information editing UOC routine does not execute any SQL statements, do not specify `HiRDB` libraries in the link command used to create executable files.
If this rule is violated, the `KFRB03094-W` message will be output.
The subsequent processing depends on the `hirdb_audit_trail` operand value in the import system definition.
- To avoid any unauthorized accesses from the import information editing UOC routine, the Datareplicator administrator must ensure the following:
 - The import information editing UOC routine does not allow unauthorized access to the databases.
 - The import information editing UOC routine does not allow tampering by third parties with malicious intent.
 - The owner and access permissions of the import information editing UOC

routine are set as shown in the following table.

Table 8-22: Owner and access permissions for an import information editing UOC routine

Owner and access permission		Information to be specified
Owner	User ID	HiRDB Datareplicator administrator
	Group ID	HiRDB administrator's group ID
Access permission	Owner	r-x (read and execute permissions)
	Group	--- (Not accessible)
	Other	--- (Not accessible)

Because there is no permission to write files, if you re-create the executable files of the import information editing UOC routine, delete the existing executable files, and then create the new executable files.

(8) Notes about linking data in a table for which a referential constraint has been defined

A table for which a referential constraint has been defined cannot be specified in an import information editing UOC routine.

(9) Notes about the extended syslog facility

If you will be using the extended syslog facility, re-link the UOC routine by using the linkage procedure described in *8.1.2 Creating an import information editing UOC routine (UNIX)*.

If the UOC routine is not linked as instructed, the extended syslog facility will not be applicable to `syslogfile` that is output for the import information editing UOC routine.

8.1.7 Sample import information editing UOC routine

This section explains the sample UOC routine that is stored in the target Datareplicator.

(1) Storage directory and filename

When you install the target Datareplicator, the sample file is stored in the following directory:

Directory

UNIX Datareplicator: `/opt/hirdbds/lib/sample/`

Windows Datareplicator: `Datareplicator-installation-directory\sample\`

Filename

hds_redit.ec

(2) Extraction environment

(a) Table subject to extraction processing

The following figure shows the structure of the sample table that is subject to extraction processing.

Figure 8-7: Structure of the sample table subject to extraction processing

Column data types	INTEGER	CHAR (30)	CHAR (2)	INTEGER	INTEGER
Column names	PCODE	PNAME	COLOR	PRICE	SQTY
	:	:	:	:	:

Note:

In this sample, only SQTY is subject to update processing.

(b) Details about extraction processing

This sample executes the following extraction processing:

- Name of update information: STOCK
- Columns subject to extraction processing: All columns
- Order of extraction: Same order as the columns
- Mapping key: PCODE column

(3) Import environment

(a) Contents of the table subject to import processing

The following figure shows the structure of the sample table that is subject to import processing. The name of the target table is STOCK.

Figure 8-8: Structure of the target table (STOCK)

Column data types	INTEGER	CHAR (30)	INTEGER	INTEGER
Column names	PCODE	PNAME	PRICE	SQTY
	:	:	:	:

(b) Import definition

The import definition is shown below. This import definition is based on the following conditions:

- All columns are imported from the source table, except the COLOR column.
- The UOC routine's name is /users/replicator/sampleuoc.

```
format STOCK
NAME PCODE      /* product code      */
NAME PNAME      /* product name      */
NAME COLOR      /* color              */
NAME PRICE      /* price              */
NAME SQTY       /* stock quantity    */
load PCODE,PNAME,PRICE,SQTY from STOCK by '/users/replicator/
sampleuoc';
```

(4) Details about processing

The editing start instruction function (`hds_ubegin`) executes `CONNECT` processing on the target HiRDB. The editing and processing instruction function (`hds_uedit`) executes `INSERT`, `UPDATE`, `DELETE`, `PURGE`, and `COMMIT`, as appropriate to the update type.

If `UPDATE` or `DELETE` results in an error (`SQLCODE 100`) because there is no row subject to update processing in the target table, the UOC routine ignores the erroneous update information and continues import processing. If any other SQL execution error occurs, the UOC routine executes `ROLLBACK` processing using the editing termination instruction function (`hds_uend`). The editing termination instruction function (`hds_uend`) executes `ROLLBACK` as required, and then executes `DISCONNECT`.

(a) Editing start instruction function (`hds_ubegin`)

The editing start instruction function (`hds_ubegin`) sets the authorization identifier and password from its `UINTERFACE_BLK` parameters into the embedded variables and issues `CONNECT` to the target HiRDB using these embedded variables as parameters.

(b) Editing and processing instruction function (`hds_uedit`)**SQL execution processing**

The editing and processing instruction function (`hds_uedit`) issues the SQL statement specified for the update type in its `UREFLECT_BLK` parameter to the target HiRDB.

```
INSERT
```

The function obtains the update data address of each column in the target table on the basis of the import data block (`UDATA_BLK`), which is the `hds_uedit` function's parameter, and then sets each data in the embedded variable. The function then executes `INSERT` using the embedded variable as the parameter.

UPDATE

The function obtains the update data address of `PCODE`, which is the mapping key column, on the basis of the key information block, which is the `hds_uedit` function's parameter, and then sets the update data in the embedded variable. Because only the `SQTY` column in the source table is subject to update processing, the import data block, which is the `hds_uedit` function's parameter, contains the update information for the `SQTY` column. The function sets the update data for this `SQTY` column in the embedded variable.

The function executes `UPDATE` using the embedded variable for the `PCODE` column as the search condition value and the embedded variable for the `SQTY` column as the update value.

DELETE

The function obtains the update data address of `PCODE`, which is the mapping key column, on the basis of the key information block, which is the `hds_uedit` function's parameter, and then sets the update data in the embedded variable. The function then executes `DELETE` using the embedded variable as the search condition.

PURGE

The function executes `PURGE TABLE` specifying `STOCK` as the name of the target table.

COMMIT

The function executes `COMMIT WORK`.

Function termination processing

This function determines the return code and status code on the basis of the `SQLCODE` SQL execution result. The following figure shows the return codes and status codes based on the `SQLCODE`s.

Table 8-23: Return codes and status codes based on the SQL codes

SQLCODE	SQL status	Return code	Status code
0	Normal termination	HDS_RET_NORM (normal termination)	N/A
100	Error indicating that the specified row was not found in the target table during <code>UPDATE</code> or <code>DELETE</code> processing	HDS_RET_WARN (minor error)	SQLCODE
Other [#]	Error resulting in cancellation of import processing	HDS_RET_ERR (serious error)	SQLCODE

#: The function sets -1 in inherited information 2 in the interface block (UINTERFACE_BLK), which is the hds_uedit function's parameter, as the flag for execution of ROLLBACK.

(c) Editing termination instruction function (hds_uend)

The editing termination instruction function (hds_uend) executes ROLLBACK and DISCONNECT. It executes ROLLBACK only when inherited information 2 is -1 in the interface block (UINTERFACE_BLK), which is the hds_uend function's parameter.

8.2 Column data editing UOC routine

A *column data editing UOC routine* that you create can be used to edit desired source column data and import it into a target database.

Notes about creating and editing a UOC routine with the IPF version of Datareplicator

Preprocess, compile, and link a created UOC routine in the 64-bit environment. Datareplicator does not support use of 32-bit libraries for executable files. If you attempt to use such a UOC routine, operation is not guaranteed.

In the IPF version, replace the `long` data type that appears in this chapter with the `int` data type.

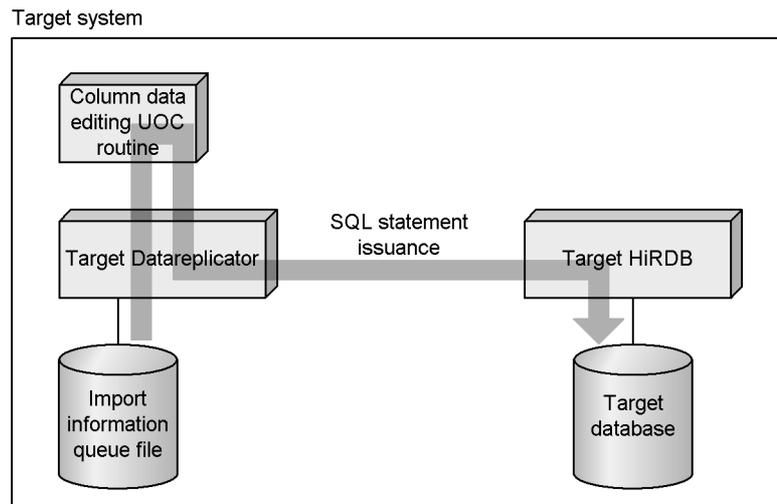
Notes about creating a UOC with non-IPF versions of Datareplicator

Preprocess, compile, and link a created UOC routine in the 32-bit environment. Datareplicator does not support use of 64-bit libraries for executable files. If you attempt to use such a UOC routine, the results might not be correct.

8.2.1 Overview of a column data editing UOC routine

When you have created a column data editing UOC routine, control is passed to the UOC routine before Datareplicator imports extraction data information into the target database. When control is returned from the UOC routine, Datareplicator imports the data edited by the UOC routine into the target database. The following figure shows the procedure for importing update information when a column data editing UOC routine is created.

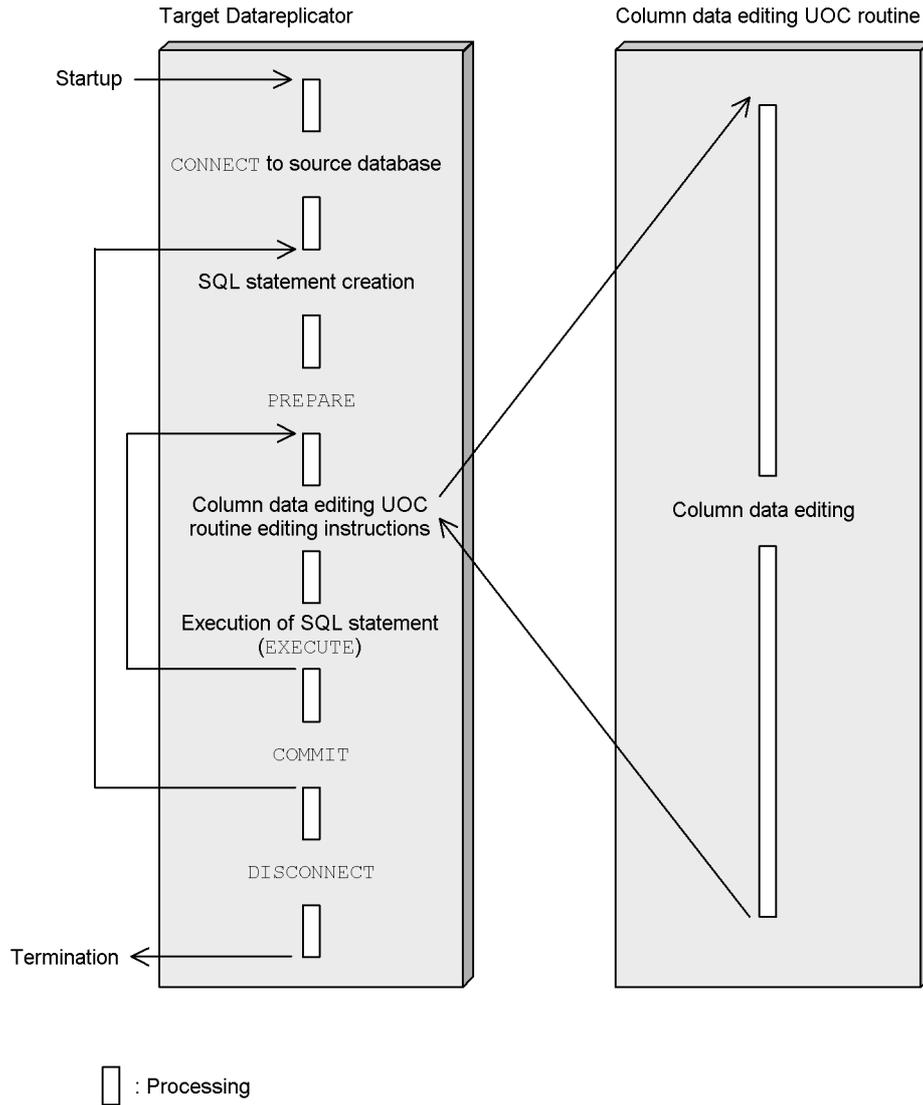
Figure 8-9: Procedure for importing update information using a column data editing UOC routine



(1) Column data editing UOC routine call timing

A column data editing UOC routine is called by Datareplicator when `INSERT` processing occurs on a table containing a source column that is subject to processing by the column data editing UOC routine and when `UPDATE` import processing occurs on a target column that is subject to processing by the column data editing UOC routine. The following figure shows the flow of control during processing by a column data editing UOC routine.

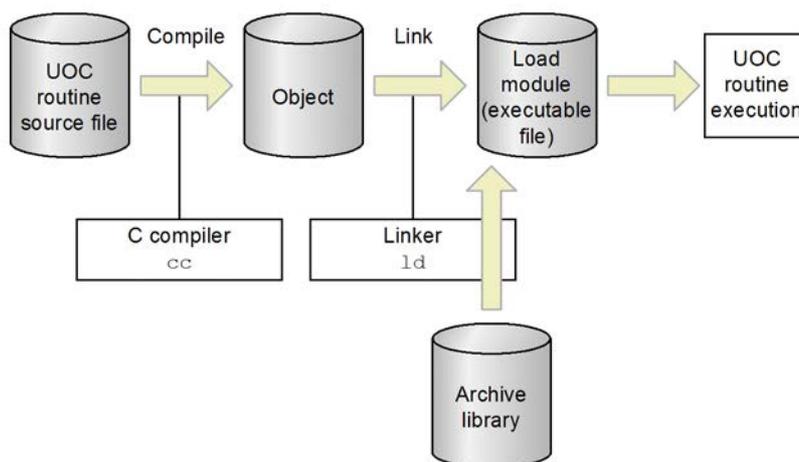
Figure 8-10: Flow of control during processing by a column data editing UOC routine



8.2.2 Column data editing UOC routine creation procedure (UNIX)

This section explains the procedure for creating a column data editing UOC routine with the UNIX Datareplicator. The following figure shows the execution procedure for a column data editing UOC routine.

Figure 8-11: Column data editing UOC routine's execution procedure

**(1) Creation**

To create a shared library for the column data editing UOC routine interface:

1. Select a target column that is to use the column data editing UOC routine interface.
2. Create the function for the column data editing UOC routine interface.
3. Compile and link the function for the column data editing UOC routine interface.

(2) Compiling and linking

The compiling and linking procedures are explained for each OS.

If an unresolved link error occurs, specify all libraries required during linking.

(a) HP-UX Datareplicator**UOC routine compilation**

```
cc -c +z -I/opt/hirdbds/include
    UOC-routine-source-filename [ UOC-routine-source-filename ] . . .
```

Compilation of a UOC routine in an environment for the IPF version

```
cc -c +DD64 +z -Y -I/opt/hirdbds/include
    UOC-source-file-name [ UOC-source-file-name ] . . .
```

-c: Object creation option.

+z: Position-independent code creation option.

-I: Directory that stores the header files provided by Datareplicator (always /opt/hirdbds/include). You can specify multiple directories if the UOC

routine requires other header files.

UOC routine linkage

```
ld -b -o libhdscuoc.sl
UOC-routine-object-filename [ UOC-routine-object-filename ] . . .
```

Linkage of a UOC routine in an environment for the IPF version

```
ld -b -o libhdscuoc.so
UOC-object-file-name [ UOC-object-file-name ] . . .
```

(b) Solaris Datareplicator

UOC routine compilation

```
/opt/SUNWspro/bin/cc -c -KPIC -I/opt/hirdbds/include
UOC-routine-source-filename [ UOC-routine-source-filename ] . . .
```

-c: Object creation option.

-KPIC: Position-independent code creation option.

-I: Directory that stores the header files provided by Datareplicator (always /opt/hirdbds/include). You can specify multiple directories if the UOC routine requires other header files.

UOC routine linkage

```
/opt/SUNWspro/bin/cc -G -o libhdscuoc.so
UOC-routine-object-filename [ UOC-routine-object-filename ] . . .
```

(c) AIX Datareplicator

UOC routine compilation

```
xlc -c -I /opt/hirdbds/include
UOC-routine-source-file-name [ UOC-routine-source-file-name ] . . .
```

-c: Object creation option

-I: Directory that stores the header files provided by Datareplicator (always /opt/hirdbds/include). You can specify multiple directories if the UOC routine requires other header files.

UOC routine linkage

```
xlc -bM:SRE -bnoentry
-bE:/opt/hirdbds/lib/libhdscuoc.exp -o libhdscuoc.a
UOC-routine-object-file-name [ UOC-routine-object-file-name ] . . .
```

(d) Linux Datareplicator

Compilation of a UOC routine in an environment for the 32-bit or IPF version

```
gcc -c -fPIC -I/opt/hirdbds/include
UOC-routine-source-filename [ UOC-routine-source-filename ] . . .
```

Compilation of a UOC routine in an environment for the EM64T version

```
gcc -c -m32 -fPIC -I/opt/hirdbds/include
UOC-source-file-name[ UOC-source-file-name ] . . .
```

-c: Object creation option.

-fPIC: Position-independent code creation option.

-I: Directory that stores the header files provided by Datareplicator (always /opt/hirdbds/include). You can specify multiple directories if the UOC routine requires other header files.

Linkage of a UOC routine in an environment for the 32-bit edition

```
ld -G -o libhdscuoc.so
UOC-routine-object-filename[ UOC-routine-object-filename ] . . .
```

Linkage of a UOC routine in an environment for the IPF version

```
gcc -shared -o libhdscuoc.so
UOC-object-file-name[ UOC-object-file-name ] . . .
```

Linkage of a UOC routine in an environment for the EM64T version

```
gcc -m32 -shared -o libhdscuoc.so
UOC-object-file-name[ UOC-object-file-name ] . . .
```

(3) Notes

You can store a created send data UOC routine library in any directory. However, do not replace `libhdscuoc.sl` (`libhdscuoc.so` for the Solaris, Linux, or HP-UX (IPF) Datareplicator and `libhdscuoc.a` for AIX Datareplicator) under `/opt/hirdbds/lib`.

8.2.3 Column data editing UOC routine creation procedure (Windows)

This section explains the procedure for creating a column data editing UOC routine with Windows Datareplicator. A UOC routine for Windows Datareplicator is created as a DLL file (`xxx.dll`).

(1) Creating the routine

In the Windows edition, you can create a column data editing UOC routine in the following development environments:

- Microsoft Visual C++ Version 5.0 or later (preprocessor: 32-bit)
- Platform SDK February 2003 or later (preprocessor: IPF)
- Visual Studio 2005 or later (preprocessor: EM64T)

To create a DLL file:

1. Select the function for your column data editing UOC routine; create this function in the C language that is used to create application programs for the Windows HiRDB. For the DLL filename for a column data editing UOC routine, use `hdscuoc.dll`. Datareplicator's UOC header files are stored in `\include` under the Datareplicator installation directory.
2. Compile and link the created function to create a DLL file. To call the UOC routine from Datareplicator, you must create an interface function with the `_ _cdecl` calling convention, and then export it.

(2) *Compiling and linking*

This subsection explains how to compile and link a UOC routine in each development environment.

(a) **Microsoft Visual C++ Version 5.0**

If you use Microsoft Visual C++ Version 5.0 to compile and link your UOC routine, select **Settings** from the **Project** menu to specify options.

The following table lists and describes the items to be specified in **Project Settings** or **Settings**.

Table 8-24: Items to be specified in Project Settings or Settings in Microsoft Visual C++ Version 5.0

Item	Category	Category setting	Setting
Compiler	Code Generation	Struct Member Alignment	8 bytes
		Runtime Library	Multi-threaded DLL
		Calling Convention	<code>_cdecl</code> *
	Preprocessor	Include file path	<code>Datareplicator-installation-directory\include</code>

If an unresolved link error occurs, specify all libraries required during linking.

(b) Platform SDK February 2003

- UOC routine compilation

```
cl.exe /c /D_MT /D_DLL /MD /D"WINVER=0x0400" /D"_WIN32_WINNT=0x0333"
/D"_WINNT" /D"NDEBUG" /D"_WINDOWS" /D"WIN32" /D"WIN64" /nologo
/I Datareplicator-installation-directory\include
UOC-source-file-name{UOC-source-file-name}...
```

- UOC routine linkage

```
link.exe /subsystem:console /incremental:no /machine:IA64 /out:hdscuoc.dll
/DLL /DEF:definition-file UOC-object-file-name{UOC-object-file-name}...
```

(c) Visual Studio 2005

If you use Visual Studio 2005 to compile and link your UOC routine, select **Settings** from the **Project** menu to specify options.

The following table shows the items to be specified.

Table 8-25: Items to be specified in Project Settings or Settings in Visual Studio 2005

Item	Category	Category setting	Setting
Platform	--	--	Win32
Configuration Properties	General	Common Language Runtime Support	No Common Language Runtime Support
Compiler	Code Generation	Struct Member Alignment	8 bytes
		Runtime Library	Multi-threaded DLL
	Advanced	Calling Convention	_cdecl *
	General	Additional Include Directories	Datareplicator-installation-directory\include

Legend:

--: Not applicable

If an unresolved link error occurs, specify all libraries required during linking.

(3) Notes on creating a column data editing UOC routine

- Check that the created UOC routine does not use the same memory mapped file that is used by the target Datareplicator or by any other system.
- If the UOC routine's processing results in an error, the name of the DLL file is displayed in the error log file as the program name.

8.2.4 Syntax for the function used with a column data editing UOC routine

You use C language to program a column data editing UOC routine. The target Datareplicator provides a function for use in creating your column data editing UOC routine. The following table shows the function that can be used for a column data editing UOC routine and provides an overview of the function.

Table 8-26: Overview of the function provided for creating a column data editing UOC routine

Function name	Function
hds_ucoleditleX() (Edit column data)	Receives column data to be edited and stores the data obtained after editing. The user is responsible for editing the data, and then storing the data obtained after editing.

(1) hds_ucoleditleX (edit column data)

A column data editing UOC routine uses the hds_ucoleditleX function (X: 1-8). You can create eight UOC functions per Datareplicator system.

(a) Function format

```
#include<hds_ucommon.h>
#include<hds_ureflect.h>
int hds_ucoleditleX(UCOLENV_BLK
*column-data-editing-interface-environment-block,
UCOLUMN_BLK *pre-edit-import-column-information,
UCOLUMN_BLK *post-edit-import-column-information,
long *status);
```

(b) Explanation of the parameters

The parameters are explained below. The function's structure is defined in the column data editing UOC routine's header files. Do not update an area whose structure is defined by the caller. For details about the header files, see *Table 8-33 Header files for the column data editing UOC*.

UCOLENV_BLK

This block stores environment information for the column data editing interface. The following table shows the contents of UCOLENV_BLK.

Table 8-27: Contents of UCOLENV_BLK

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
interface_blk	56	UINTERFACE_BLK	Interface block	Caller	Import system definition information
reflect_blk	96	UREFLECT_BLK	Import control block	Caller	Source database update information in import data
auth_id	8	char	Authorization identifier	Caller	Authorization identifier used for importing into the target database
tbl_id	30	char	Table identifier	Caller	Table identifier used for importing into the target database
reserve	2	char	Reserved	Caller	N/A

UINTERFACE_BLK, UREFLECT_BLK

The structures of UINTERFACE_BLK and UREFLECT_BLK are the same as for an import information editing UOC routine, but their contents and usage are different. Table 8-28 *Contents of UINTERFACE_BLK* shows the contents of UINTERFACE_BLK, and Table 8-29 *Contents of UREFLECT_BLK* shows the contents of UREFLECT_BLK.

Table 8-28: Contents of UINTERFACE_BLK

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
auth_id	8	char	Authorization identifier	Caller	Authorization identifier specified in the import system definition
password	30	char	Password	Caller	Password specified in the import system definition

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
exsysid	2	char	Source Datareplicator identifier	Caller	Source Datareplicator identifier specified in the source Datareplicator's extraction system definition, expressed as a string of lowercase hexadecimal characters. During data linkage with XDM/DS, 00 is set.
repid1	2	char	Identifier 1	Caller	Data linkage identifier specified in the import system definition
repid2	2	char	Identifier 2	Caller	Target Datareplicator identifier specified in the import system definition
*inherinf1	4	char *	N/A	N/A	Not used with a column data editing UOC function.
inherinf2	4	long	N/A	N/A	Not used with a column data editing UOC function.
stopinf	4	long	N/A	N/A	Not used with a column data editing UOC function.

Table 8-29: Contents of UREFLECT_BLK

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
updkind	8	char	Update type	Caller	Type of update processing on the source database (as an 8-byte character string, as shown below; if the type is shorter than 8 bytes, spaces are added): INSERT INSERT statement UPDATE UPDATE statement
auth_id	30	char	Authorization identifier	Caller	Authorization identifier used to update the source database
tbl_id	30	char	Table identifier	Caller	Table identifier used to update the source database

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
upd_date	4	char	Update date	Caller	Date the source database was updated (date the update information was output to the journal). The date is displayed in unsigned packed decimal format (YYYYMMDD).
upd_time	4	char	Update time	Caller	Time the source database was updated (time the update information was output to the journal). The time is displayed in unsigned packed decimal format (HHMMSS.TT where TT is 1/100 sec.).
uapname	8	char	Source UAP name	Caller	Name of the user application program used to update the source database
extsysid	4	char	Source database's system ID	Caller	When the source database is a HiRDB: Source database's HiRDB identifier When the source database is a mainframe (XDM/DS, PDM2 E2, or RDB1 E2): Source database's subsystem ID
infflag1	1	unsigned char	Flag 1	Caller	Not used. HDS_NULL_FLAG (0x00) Fixed value
infflag2	1	unsigned char	Flag 2	Caller	Not used. HDS_NULL_FLAG (0x00) Fixed value
infflag3	1	unsigned char	Flag 3	Caller	Not used. (0x00) Fixed value
reserve1	1	char	N/A	N/A	(For boundary adjustment)
infflag4	4	long	Flag 4	Caller	Not used. (0x00) Fixed value

UCOLUMN_BLK

This block stores information about the column data. For the pre-edit import column information, the caller stores information about the update data in the update information and passes it to the column data editing UOC routine. For the post-edit import column information, the information edited by the UOC routine is stored. The UOC routine must store the data type, column option (optional), column data length, and data obtained after editing in the area indicated by the data address in the post-edit import column information. The following table shows the contents of UCOLUMN_BLK.

Table 8-30: Contents of UCOLUMN_BLK

Member name	Length (bytes)	Attribute	Area name	Pre-edit Defined by	Post-edit Defined by	Description
colname	30	char	Column name	Caller	Caller	Pre-edit import column information: Source table's column name is set. Post-edit import column information: Target table's column name is set.
coltype	1	unsigned char	Column type	Caller	Caller	Source column's column data type is set as the pre-edit import column information. After editing, set the column data type as the post-edit import column information. Use the same mnemonic as the import information editing interface (BLOB, BINARY, UNPACK, and ADT (SGMLTEXT, FREEWORD, XML) data type cannot be specified for the post-edit import column information).

Member name	Length (bytes)	Attribute	Area name	Pre-edit Defined by	Post-edit Defined by	Description
coloption	1	unsigned char	Column option	Caller	Caller	Source column's column option is set as the pre-edit import column information. After editing, set the column option as the post-edit import column information. The following options can be set: HDS_COL_NOOPT (0x00): No option HDS_COL_NULL (0x01): Null data HDS_COL_ARRAY (0x04): Array column (the array column option is not permitted in the post-edit import column information)
elementnum	4	union	Valid element number in array column	Caller	N/A	If an array is specified as the column option in the pre-edit import column information, the valid element number of the array column is set (this information cannot be set in the post-edit import column information).
mltcolkind	1	unsigned char	N/A	N/A	N/A	Not used with a column data editing UOC routine.
adtfunc	1	unsigned char	ADT column import method	Caller	N/A	One of the following values is set as the abstract data type linkage method: 0x00 Other than abstract data type HDS_ADTRREP_NORM (0x01) Constructor function import method

8. User Own Coding Routines

Member name	Length (bytes)	Attribute	Area name	Pre-edit Defined by	Post-edit Defined by	Description
charset	1	char	Character set type	Caller	N/A	<p>Character set type of the column data that is to be passed to the UOC routine</p> <p>Do not change the character set type before or after editing. If the character set type is changed, Datareplicator ignores the edited character set type and resumes processing using the character set in effect before editing.</p> <p>When no character set is specified:</p> <p>X'00' (HDS_CSET_DEFAULT)</p> <p>When a character set (EBCDIK) is specified:</p> <p>X'01' (HDS_CSET_EBCDIK)</p>
reserve1	1	char	N/A	N/A	N/A	(For boundary adjustment)
collen	4	long	Column data length	Caller	Caller	<p>Column data length. For details, see Table 8-31 <i>Specification rules for the column data length</i>.</p> <p>If the column option is null data, 0 is set in the pre-edit import column information. If null data is set as the column option in the post-edit import column information, there is no need to set the column data length, regardless of the data type.</p>

Member name	Length (bytes)	Attribute	Area name	Pre-edit Defined by	Post-edit Defined by	Description
dataptr	4	union	Data address	Caller	N/A	Pre-edit import column information: Address of the pre-edit column data is set. If the column option is null data, NULL is set as the data address.
dataptr	4	union	Data address	Caller	N/A	Post-edit import column information: The address of the column data editing area is set. The column data editing area has a size of 32002 bytes. The column data editing UOC routine must place the edited data in the area indicated by this address. For the address type, use the same mnemonic as the import information editing interface.

Table 8-31: Specification rules for the column data length

Before/after conversion	Pre-edit import column information		Post-edit import column information		Remarks
	Specified	Specified information	Required	Specified information	
NCHAR	Yes	Pre-edit data length [characters]	Yes	Post-edit data length [characters]	N/A
CHAR MCHAR UNPACK	Yes	Pre-edit data length [bytes]	Yes	Post-edit data length [bytes]	UNPACK type is not permitted for the post-edit import column data.

Before/after conversion Column data type	Pre-edit import column information		Post-edit import column information		Remarks
	Specified	Specified information	Required	Specified information	
NVARCHAR VARCHAR MVARCHAR	No	N/A	No	N/A	Edit the pre-edit data using the real length stored in the first 2 bytes of the data area. For the post-edit data, set its real length in the first 2 bytes of the data area. If the column data type is NVARCHAR, set the length in characters; if it is VARCHAR or MVARCHAR, set the length in bytes.
DECIMAL	Yes	Leading 2 bytes Precision (total number of digits) Trailing 2 bytes Scaling (decimal places)	Yes	Leading 2 bytes Precision (total number of digits) Trailing 2 bytes Scaling (decimal places)	The maximum precision is 38 in the following cases: <ul style="list-style-type: none"> The target HiRDB version is 08-04 or later. The target DBMS is Oracle or SQL Server. In all other cases, the maximum precision is 29.
TIMESTAMP	Yes	Leading 2 bytes Data length before editing $7 + \uparrow p/2$ \uparrow [bytes] Trailing 2 bytes Scaling (decimal places)	Yes	Leading 2 bytes Data length after editing $7 + \uparrow p/2$ \uparrow [bytes] Trailing 2 bytes Scaling (decimal places)	p is a numeric value indicating the scaling (decimal places).

Before/after conversion	Pre-edit import column information		Post-edit import column information		Remarks
	Specified	Specified information	Required	Specified information	
Column data type					
Other	No	N/A	No	N/A	For the data length, see the HiRDB data format.

(c) Status

You can set a status value in the event the column data editing UOC routine returns control with an error. The specified status value will be displayed in the error message when control is returned to Datareplicator.

(d) Return value

Set the return value when control is returned to the caller. The following table shows the return values that can be set by a column data editing UOC routine.

Table 8-32: Return values from hds_ucoeditX()

Status	Code	Mnemonic	Processing after control is returned
Normal termination	0	HDS_RET_NORM	Resumes processing.
Error detected (unresumable level)	Other than 0	HDS_RET_ERR	Outputs a message at the caller, and then cancels import processing.

(2) Header files used with a column data editing UOC routine

The header files used with a column data editing UOC routine are stored in the following directory:

UNIX Datareplicator: `/opt/hirdbds/include/`

Windows Datareplicator: `Datareplicator-installation-directory\include\`

The following table lists and describes the header files for a column data editing UOC routine.

Table 8-33: Header files for the column data editing UOC

Filename	Description	Members associated with the column data editing UOC routine
<code>hds_ucommon.h</code>	Structures and mnemonic codes common to column data editing UOC routines	<ul style="list-style-type: none"> • <code>UINTERFACE_BLK</code> • Mnemonics for the return value

Filename	Description	Members associated with the column data editing UOC routine
hds_ureflect.h	Structure and mnemonics for the column data editing UOC routine	<ul style="list-style-type: none"> • UCOLENV_BLK • UREFLECT_BLK • Mnemonics for UREFLECT_BLK • UCOLUMN_BLK • Mnemonics for the data type • Mnemonics for the data address

8.2.5 Notes on creating a column data editing UOC routine

To execute a column data editing UOC routine, specify the directory path to the shared library file in an environment variable. Note that the environment variable and the shared library file to be used depend on the OS. The following table shows the environment variable and shared library file for each OS.

Table 8-34: Environment variable and shared library file for each OS

OS	Environment variable	Shared library file
HP-UX (32-bit or 64-bit edition)	SHLIB_PATH	libhdscuoc.sl
HP-UX (IPF version), Solaris, or Linux (32-bit, IPF, or EM64T version)	LD_LIBRARY_PATH	libhdscuoc.so
AIX	LIBPATH	libhdscuoc.a

If the environment variable is not specified or there is no shared library file at the directory path specified in the environment variable, Datareplicator executes import processing without using the column data editing UOC routine.

(1) Limitations on signals

Do not conduct signal operations in a column data editing UOC routine's function.

(2) Limitations on file manipulations

You can manipulate only user-specific files (that are open) within a column data editing UOC routine's function. The following are the file-related rules:

- Do not use the standard input, standard output, or standard error output.
- Do not read, write, or close any file that is not opened by the UOC routine.
- Do not manipulate any file that belongs to HiRDB or Datareplicator.

(3) Limitations on shared memory

Do not use the same shared memory that is used by Datareplicator or any other system.

(4) Limitations on data types

- Columns of the following data types are not supported by column data editing UOC routines; if such a column is specified, an error occurs during import definition analysis:
 - Abstract data types (SGMLTEXT, FREEWORD, XML)
 - BLOB type
 - BINARY type
- SMALLFLOAT-type data is passed as FLOAT-type data to the UOC routine (to avoid overflow that might occur due to a difference in representation range).
- If an extracted column is the UNPACK type, Datareplicator converts it to DECIMAL-type data, and then passes it to the column data editing UOC routine. If Datareplicator cannot convert the UNPACK data to DECIMAL data, it passes the UNPACK data to the UOC routine.

(5) Limitations on SQL execution

Do not execute any SQL statement within or as an extension of the column data editing UOC routine's function. If executed, transaction management by Datareplicator might become invalid.

(6) Notes about the AIX Datareplicator

For the AIX Datareplicator, create all functions from `hds_ucoedit1()` through `hds_ucoedit8()`; otherwise, import processing will result in an error.

(7) Notes about handling repetition columns

A column data editing UOC routine passes each element of the data (a single call to a column data editing UOC function passes one element data item to this UOC function). This means that a column data editing UOC routine cannot be used to increase or decrease the number of elements.

8.2.6 Sample column data editing UOC routine

This section explains the sample UOC routine that is stored in the target Datareplicator.

(1) Storage directory and filename

Directory

UNIX Datareplicator: `/opt/hirdbds/lib/sample/`

Windows Datareplicator: `Datareplicator-installation-directory\sample\`

Filename

`hds_cedit.c`

8.3 Send data UOC routine

A *send data UOC routine* that you create can be used before extracted update information is sent to the target system to determine whether the information is to be imported. You use the source Datareplicator to create a send data UOC routine.

Notes about creating and editing a UOC routine with the IPF version of Datareplicator

In the IPF version, you must preprocess, compile, and link your created UOC routine in the 64-bit environment. Datareplicator does not support use of 32-bit libraries for executable files. If you attempt to use such a UOC routine, operation is not guaranteed.

In the IPF version, replace the `long` data type that appears in this chapter with the `int` data type.

Notes about creating a UOC routine with a non-IPF version of Datareplicator

Even if you are using the 64-bit Datareplicator, you must preprocess, compile, and link a created UOC routine in the 32-bit environment. Datareplicator does not support use of 64-bit libraries for executable files. If you attempt to use such a UOC routine, the results might not be correct.

8.3.1 Overview of a send data UOC routine

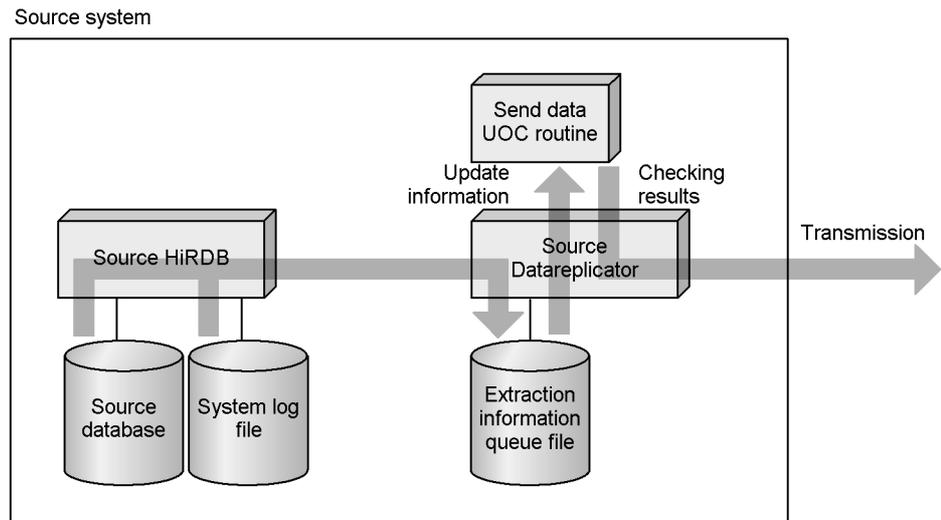
When you have created a send data UOC routine, control is passed to the UOC routine before the source Datareplicator sends the extracted update information to the target system. When control is returned from the UOC routine, Datareplicator sends the update information to the target system in accordance with the checking results.

The following data is passed to a send data UOC routine:

- Data that satisfies conditions specified in the *where* clause of the *send* statement in the extraction definition
- Data determined to be subject to transmission as a result of loopback suppression checking

Datareplicator passes update information to the send data UOC routine in units of the tables that are subject to extraction processing, as specified in the extraction definition. If multiple sets of update information are to be extracted from a single table, Datareplicator merges the update information into a single set, and then passes it to the UOC routine. For a source Datareplicator to be able to use a send data UOC routine, you must specify *use* in the *senduoc* operand in the transmission environment definition. The following figure shows the procedure for sending update information when a send data UOC routine is created.

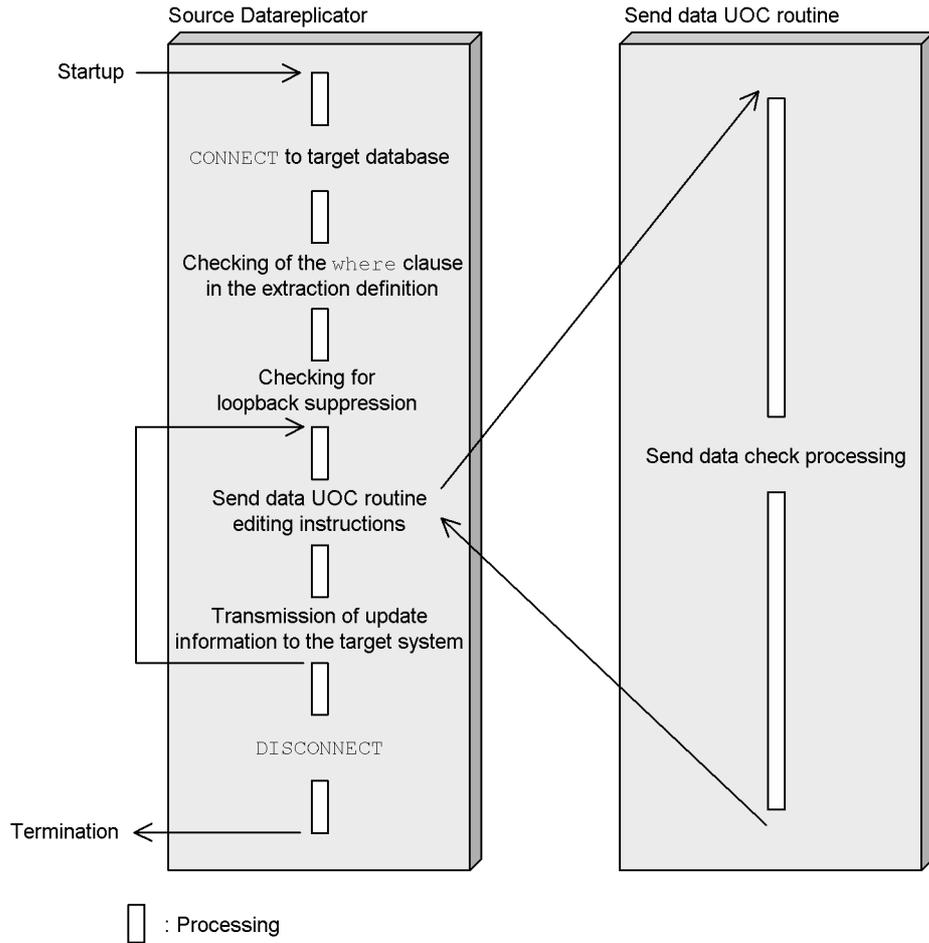
Figure 8-12: Procedure for sending update information using a send data UOC routine



(1) Send data UOC routine call timing

A send data UOC routine is called by Datareplicator when checking of the where clause in the extraction definition and checking for loopback suppression are completed. The following figure shows the flow of control during processing by a send data UOC routine.

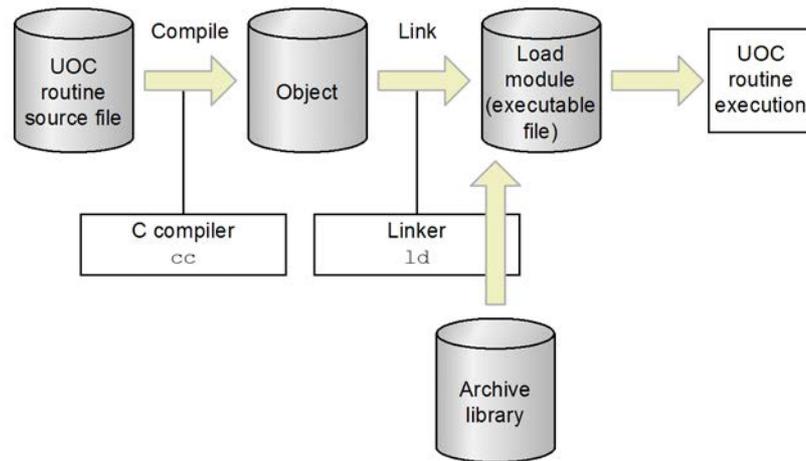
Figure 8-13: Flow of control during processing by a send data UOC routine



8.3.2 Send data UOC routine creation procedure (UNIX)

This section explains the procedure for creating a send data UOC routine with UNIX Datareplicator. The following figure shows the procedure up to the point where a send data UOC routine is executed.

Figure 8-14: Send data UOC routine's execution procedure

**(1) Creation**

To create a shared library for the send data UOC routine:

1. Select the update information that is to be checked by the send data UOC routine.
2. Create the function for the send data UOC routine.
3. Compile and link the function for the send data UOC routine.

(2) Compiling and linking

The compiling and linking procedures are explained for each OS.

If an unresolved link error occurs, specify all libraries required during linking.

(a) HP-UX Datareplicator

- **UOC routine compilation**

```
cc -c +z -I/opt/hirdbds/include
  UOC-routine-source-filename[ UOC-routine-source-filename]...
```

- **Compilation of a UOC routine in an environment for the IPF version**

```
cc -c +z -Y -I/opt/hirdbds/include UOC-source-file-name[ UOC-source-file-name]...
```

-c: Object creation option.

+z: Position-independent code creation option.

-I: Directory that stores the header files provided by Datareplicator (always /opt/hirdbds/include). You can specify multiple directories if the UOC routine requires other header files.

- **UOC routine linkage**

```
ld -b -o libhdesuoc.sl UOC-routine-object-filename[UOC-routine-object-filename]...
```

- **Linkage of a UOC routine in an environment for the IPF version**

```
ld -b -o libhdesuoc.so UOC-object-file-name[UOC-object-file-name]...
```

(b) Solaris Datareplicator

- **UOC routine compilation**

```
/opt/SUNWspro/bin/cc -c -KPIC -I/opt/hirdbds/include
  UOC-routine-source-filename[UOC-routine-source-filename]...
```

-c: Object creation option.

-KPIC: Position-independent code creation option.

-I: Directory that stores the header files provided by Datareplicator (always /opt/hirdbds/include). You can specify multiple directories if the UOC routine requires other header files.

- **UOC routine linkage**

```
/opt/SUNWspro/bin/cc -G -o libhdesuoc.so
  UOC-routine-object-filename[UOC-routine-object-filename]...
```

(c) AIX Datareplicator

- **UOC routine compilation**

```
xlc -c -I /opt/hirdbds/include
  UOC-routine-source-file-name[UOC-routine-source-file-name]...
```

-c: Object creation option

-I: Directory that stores the header files provided by Datareplicator (always /opt/hirdbds/include). You can specify multiple directories if the UOC routine requires other header files.

- **UOC routine linkage**

```
xlc -bM:SRE -bnoentry
      -bE:/opt/hirdbds/lib/libhdesuoc.exp -o libhdesuoc.a
      UOC-routine-object-file-name[UOC-routine-object-file-name]...
```

(d) Linux Datareplicator

- **Compilation of a UOC routine in an environment for the 32-bit or IPF version**

```
gcc -c -fPIC -I/opt/hirdbds/include
      UOC-routine-source-file-name[UOC-routine-source-file-name]...
```

- **UOC routine compilation in an environment for the EM64T version**

```
gcc -c -m32 -fPIC -I/opt/hirdbds/include UOC-source-file-name[UOC-source-file-name]...
```

-c: Object creation option.

-fPIC: Position-independent code creation option.

-I: Directory that stores the header files provided by Datareplicator (always /opt/hirdbds/include). You can specify multiple directories if the UOC routine requires other header files.

- **Linkage of a UOC routine in an environment for the 32-bit edition**

```
ld -G -o libhdsuoc.so UOC-routine-object-filename[UOC-routine-object-filename]...
```

- **Linkage of a UOC routine in an environment for the IPF version**

```
gcc -shared -o libhdesuoc.so UOC-object-file-name[UOC-object-file-name]...
```

- **Linkage of a UOC routine in an environment for the EM64T version**

```
gcc -m32 -shared -o libhdesuoc.so UOC-object-file-name[UOC-object-file-name]...
```

(3) Notes

You can store a created send data UOC routine library in any directory. However, do not replace `libhdesuoc.sl` (`libhdesuoc.s` for the Solaris or Linux Datareplicator, and `libhdesuoc.exp` for AIX Datareplicator) under `/opt/hirdbds/lib`.

Before executing the UOC, you must specify the directory containing the send data UOC routine library (`libhdesuoc.sl` or `libhdesuoc.so`) in the `node_shlibpath` operand in the extraction system definition. If the send data UOC routine library is not found in the directory specified in the `node_shlibpath` operand, Datareplicator sends all update information.

8.3.3 Send data UOC routine creation procedure (Windows)

This section explains the procedure for creating a send data UOC routine with Windows Datareplicator. A UOC routine for Windows Datareplicator is created as a DLL file (`xxx.dll`).

(1) How to create

In the Windows edition, you can create a send data UOC routine in the following development environments:

- Microsoft Visual C++ Version 5.0 or later (preprocessor: 32-bit)
- Platform SDK February 2003 or later (preprocessor: IPF)
- Visual Studio 2005 or later (preprocessor: EM64T)

To create a DLL file:

1. Select the function for your send data UOC routine; create this function in the C language that is used to create application programs for Windows HiRDB. Use `senduoc.dll` as the DLL filename for a send data UOC routine. Datareplicator's UOC routine header files are stored in `\include` under the Datareplicator installation directory.
2. Compile and link the created function to create a DLL file. To call the UOC routine from Datareplicator, you must create an interface function with the `__cdecl` calling convention, and then export it.

(2) Compiling and linking

This subsection explains how to compile and link a UOC routine in each development environment.

(a) Microsoft Visual C++ Version 5.0

If you use Microsoft Visual C++ Version 5.0 to compile and link your UOC routine, select **Settings** from the **Project** menu to specify options.

The following table lists and describes the items to be specified in **Project Settings** or **Settings**.

Table 8-35: Items to be specified in Project Settings or Settings in Microsoft Visual C++ Version 5.0

Item	Category	Category setting	Setting
Compiler	Code Generation	Struct Member Alignment	8 bytes
		Runtime Library	Multi-threaded DLL
		Calling Convention	_cdecl *
	Preprocessor	Include file path	Datareplicator-installation-directory\include

If an unresolved link error occurs, specify all libraries required during linking.

(b) Platform SDK February 2003

- UOC routine compilation

```
cl.exe /c /D_MT /D_DLL /MD /D"WINVER=0x0400" /D"_WIN32_WINNT=0x0333"
/D"_WINNT" /D"NDEBUG" /D"_WINDOWS" /D"WIN32" /D"WIN64" /nologo
/I Datareplicator-installation-directory\include
UOC-source-file-name[UOC-source-file-name]...
```

- UOC routine linkage

```
link.exe /subsystem:console /incremental:no /machine:IA64 /out: senduoc.dll
/DLL /DEF:definition-file UOC-object-file-name[UOC-object-file-name]...
```

(c) Visual Studio 2005

If you use Visual Studio 2005 to compile and link your UOC routine, select **Settings** from the **Project** menu to specify options.

The following table lists and describes the items to be specified.

Table 8-36: Items to be specified in Project Settings or Settings in Visual Studio 2005

Item	Category	Category setting	Setting
Platform	--	--	Win32
Configuration Properties	General	Common Language Runtime Support	No Common Language Runtime Support
Compiler	Code Generation	Struct Member Alignment	8 bytes
		Runtime Library	Multi-threaded DLL
	Advanced	Calling Convention	_cdecl *
	General	Additional Include Directories	<i>Datareplicator-installation-directory\include</i>

Legend:

--: Not applicable

If an unresolved link error occurs, specify all libraries required during linking.

(3) Notes

- Check that the created UOC routine does not use the same memory mapped file that is used by the source Datareplicator or by any other system.
- If the UOC routine's processing results in an error, the name of the DLL file is displayed in the error log file as the program name.
- Add the folder containing the UOC routine to the `path` system environment variable.

8.3.4 Syntax for the function used with a send data UOC routine

You use C language to program a send data UOC routine. The source Datareplicator provides a function for use in creating a send data UOC routine. The following table shows the function that can be used by a send data UOC routine and provides an overview of the function.

Table 8-37: Overview of the function provided for creating a send data UOC routine

Function name	Function
hds_usendcheck() (check send data)	Receives send data (update information) and stores the checking results. The user is responsible for specifying the information to be checked and returning the results.

(1) hds_usendcheck (check send data)

A send data UOC routine uses the hde_usendcheck() function. You can create one UOC function per source Datareplicator system.

(a) Function format

```
#include<hde_usend.h>
int hde_usendcheck(
HDE_EXT_ENVINFO *envinfo, /* (IN) transmission environment */
/* information block */
HDE_EXT_TBLINFO *tblinfo, /* (IN) extraction table */
/* information block */
HDE_EXT_COLINFO *colinfo, /* (IN) extraction data */
/* information block */
int *status); /* (OUT) transmission UOC status */
```

(b) Explanation of the parameters

The parameters are explained below. The function's structure is defined in the send data UOC routine's header file. Do not update an area whose structure is defined by the caller. For details about the header files, see *Table 8-45 Header file for a send data UOC routine*.

HDE_EXT_ENVINFO

This block stores environment information for the send data UOC routine. The following table shows the contents of HDE_EXT_ENVINFO.

Table 8-38: Contents of HDE_EXT_ENVINFO

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
sendid	9	char	Target identifier	Caller	Identifier indicting the destination of the corresponding update information (ends with NULL)

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
hdeid	1	unsigned char	Source Datareplicator identifier	Caller	Value of the hdeid operand in the extraction system definition Two-character identifier is expressed as one byte. Example: C1 → 0xC1
hdsid	1	unsigned char	Import system identifier	Caller	Value of the sendhdsid operand in the transmission environment definition Two-character identifier is expressed as one byte. Example: C1 → 0xC
dsid	1	unsigned char	Data linkage identifier	Caller	Value of the dsid operand in the transmission environment definition or the extraction environment definition (the dsid operand in the transmission environment definition takes precedence over the dsid operand in the extraction environment definition) Two-character identifier is expressed as one byte. Example: C1 → 0xC1
reserve1	20	char	Reserved	Caller	Reserved (0x00...00)

HDE_EXT_TBLINFO

The following table shows the contents of HDE_EXT_TBLINFO.

Table 8-39: Contents of HDE_EXT_TBLINFO

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
tableid	4	long	Extraction table ID	Caller	Extraction table ID of the corresponding update information
tablename	31	char	Extraction table name	Caller	Extraction table name of the corresponding update information (ends with NULL)

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
schemaname	9	char	Authorization identifier	Caller	Authorization identifier for the extraction table of the corresponding update information (ends with NULL)
exttime	8	char	Extraction time	Caller	Extraction time of the corresponding update information Output in unsigned packed decimal format (2 characters per byte) (YYYYMMDDhhmmss00)
extkind	1	unsigned char	Update type	Caller	SQL type used for updating INSERT: HDE_EXT_INSERT (0x01) UPDATE: HDE_EXT_UPDATE (0x02) DELETE: HDE_EXT_DELETE (0x03) PURGE: HDE_EXT_PURGE (0x04)
apkind	1	unsigned char	Loopback information	Caller	Loopback information 0x00: Update information from UAP 0x01: Update information from the target Datareplicator (hdesqle operand)
reserve1	11	char	Reserved	Caller	Reserved (0x00...00)

HDE_EXT_COLINFO

Table 8-40 *Contents of HDE_EXT_COLINFO* shows the contents of HDE_EXT_COLINFO, and Table 8-41 *Contents of HDE_EXT_DATAINFO* shows the contents of HDE_EXT_DATAINFO that can be referenced from HDE_EXT_COLINFO. If the value of extkind in HDE_EXT_TBLINFO is HDE_EXT_PURGE, the contents of HDE_EXT_COLINFO are the null value and cannot be referenced.

Table 8-40: Contents of HDE_EXT_COLINFO

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
colnum	4	long	Number of column information items	Caller	Number of extraction columns in the corresponding update information (number of HDE_EXT_DATAINFO entries)
*datainfo	4	HDE_EXT_DATAINFO *	Start address of extraction data information	Caller	Start address of the array of extraction data information (HDE_EXT_DATAINFO)
reserve	8	char	Reserved	Caller	Reserved

Table 8-41: Contents of HDE_EXT_DATAINFO

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
colid	4	long	Extraction column ID	Caller	Extraction column ID
colname	31	char	Extraction column name	Caller	Extraction column name (ends with NULL)
coltype	1	unsigned char	Column data type	Caller	Data type of the extraction column. For a list of column data types, see Table 8-43 <i>Column data types</i> .
colopt	1	unsigned char	Column option	Caller	Options for the extraction column Mapping key: HDE_EXT_MAPPINGKEY (0x01) Repetition column: HDE_EXT_MCOL (0x02)
charset	1	unsigned char	Character set type	Caller	Character set type for the corresponding column When no character set is specified: HDE_CSET_DEFAULT 0x00 When a character set (EBCDIK) is specified: HDE_CSET_EBCDIK 0x01

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
reserve1	2	char	Reserved	Caller	Reserved (0x00...00)
bf_dataalen	4	long	Column data length before updating ^{#1}	Caller	Data length in the extraction column before updating 0: Indicates 0-byte data or the null value. -1: Indicates that the pre-update column data is not used.
bf_dataptr	4	union ^{#2}	Column data before updating ^{#1}	Caller	Start address of the column data in the extraction column before updating 0: Indicates the null value. -1: Indicates that the pre-update column data is not used.
af_dataalen	4	long	Column data length after updating ^{#1}	Caller	Data length in the extraction column after updating 0: Indicates 0-byte data or the null value. -1: Indicates that the post-update column data is not used.
af_dataptr	4	union ^{#2}	Column data after updating ^{#1}	Caller	Start address of the column data in the extraction column after updating 0: Indicates the null value. -1: Indicates that the post-update column data is not used.
precision	2	short	Precision	Caller	If coltype is HDE_T_DEC, the precision (total length) is set; otherwise, 0 is set.

Member name	Length (bytes)	Attribute	Area name	Defined by	Description
scale	2	short	Scaling	Caller	If coltype is HDE_T_DEC, scaling (decimal places) is set; otherwise, 0 is set.
sec_preci	2	short	Decimal places for seconds	Caller	If coltype is HDE_T_TIMESTAMP, the decimal places for seconds is set; otherwise, 0 is set.
reserve2	8	char	Reserved	Caller	Reserved (0x00...00)

#1

Table 8-42 *Combination of data before and after update processing* shows the combinations of data before and after update processing. In the case of INSERT, DATAINFO (array) is sorted by the column ID; in the case of UPDATE or DELETE, it is sorted in the order extracted.

#2

If the corresponding extraction column has the CHAR or VARCHAR attribute, the character set of the corresponding extraction column is used for the data to be passed.

Table 8-42: Combination of data before and after update processing

Data type		Data length before updating	Data before updating	Data length after updating	Data after updating
INSERT	n-byte data	-1	-1	n	addr
	0-byte data	-1	-1	0	addr
	NULL data	-1	-1	0	0

Data type			Data length before updating	Data before updating	Data length after updating	Data after updating
UPDATE for non-mapping key	From <i>n</i> -byte data	To <i>n</i> -byte data	-1	-1	<i>n</i>	addr
		To 0-byte data	-1	-1	0	addr
		To NULL data	-1	-1	0	0
	From 0-byte data	To <i>n</i> -byte data	-1	-1	<i>n</i>	addr
		To 0-byte data	-1	-1	0	addr
		To NULL data	-1	-1	0	0
	From NULL data	To <i>n</i> -byte data	-1	-1	<i>n</i>	addr
		To 0-byte data	-1	-1	0	addr
		To NULL data	-1	-1	0	0
UPDATE for mapping key	From <i>n</i> -byte data	To <i>n</i> -byte data	<i>n</i>	addr	<i>n</i>	addr
		To 0-byte data	<i>n</i>	addr	0	addr
		To NULL data	<i>n</i>	addr	0	0
	From 0-byte data	To <i>n</i> -byte data	0	addr	<i>n</i>	addr
		To 0-byte data	0	addr	0	addr
		To NULL data	0	addr	0	0
	From NULL data	To <i>n</i> -byte data	0	0	<i>n</i>	addr
		To 0-byte data	0	0	0	addr
		To NULL data	0	0	0	0
DELETE [#]	<i>n</i> -byte data		<i>n</i>	addr	-1	-1
	0-byte data		0	addr	-1	-1
	NULL data		0	0	-1	-1

#

Only the mapping key data is passed. In the case of same-value updating on variable-length data or non-updating, all data information before and after updating is -1.

The following table lists the column data types.

Table 8-43: Column data types

Column data type	Mnemonic	Code
INTEGER	HDE_T_INT	0xF1
SMALLINT	HDE_T_SINT	0xF5
LARGE DECIMAL	HDE_T_DEC	0xE5
FLOAT	HDE_T_FLT	0xE1
DOUBLE PRECISION	HDE_T_DBL	0xE1
SMALL FLOAT	HDE_T_SFLT	0xE3
REAL	HDE_T_REAL	0xE3
CHARACTER	HDE_T_CHAR	0xC5
VARCHAR	HDE_T_VCHAR	0xC1
NCHAR	HDE_T_NCHAR	0xB5
NVARCHAR	HDE_T_NVCHAR	0xB1
MCHAR	HDE_T_MCHAR	0xA5
MVARCHAR	HDE_T_MVCHAR	0xA1
DATE	HDE_T_DATE	0x71
TIME	HDE_T_TIME	0x79
TIMESTAMP	HDE_T_TIMESTAMP	0x7D
INTERVAL YEAR TO DAY	HDE_T_YTD	0x65
INTERVAL HOUR TO SECOND	HDE_T_HTS	0x6F
ADT (abstract data type)	HDE_T_ADT	0x83
BLOB	HDE_T_BLOB	0x93
BINARY	HDE_T_BINARY	0x91

```
int *status
```

Indicates the UOC routine's status. This parameter is used to display the status in the event an error occurs within the UOC routine. When the function is called, this parameter is initialized to 0, and then passed back to the calling function.

The status code is embedded in the KFRB02052-E error message, which is output when the return value is neither HDE_EXT_SEND(1) nor HDE_EXT_NOSEND(0).

This embedded value has no effect on the transmission process's action.

(c) Return value

Set the return value when the send data UOC routine returns control to the caller. The following table shows the return values that can be set by a send data UOC routine.

Table 8-44: Return values from hde_usendcheck()

Status	Code	Mnemonic	Processing after control is returned
Update information is sent	1	HDE_EXT_SEND	Sends the corresponding update information to the target system.
Update information is not sent	0	HDE_EXT_NOSEND	Does not send the corresponding update information to the target system.
Transmission processing is terminated	-1	HDE_EXT_STOP	Terminates transmission to the target system. When this value is returned, the transmission process outputs the KFRB02052-E error message and cancels processing for the corresponding destination. The reason for terminating transmission processing is set in the hde_usendcheck() function's status.
	Other		

(2) Header file used with a send data UOC routine

The header file used with a send data UOC routine is stored in the following directory:

UNIX Datareplicator: /opt/hirdbds/include/

Windows Datareplicator: *Datareplicator-installation-directory*\include\

The following table shows the header file for a send data UOC routine.

Table 8-45: Header file for a send data UOC routine

Filename	Description	Members associated with the send data UOC routine
hde_usend.h	Structure and mnemonic codes for the send data UOC routine	<ul style="list-style-type: none"> • HDE_EXT_ENVINFO • HDE_EXT_TBLINFO • HDE_EXT_COLINFO • HDE_EXT_DATAINFO • Mnemonics for the data type • Mnemonics for the data address

8.3.5 Notes on creating a send data UOC routine

To execute a send data UOC routine, you must specify the directory path to the shared library file (`libhdesuoc.sl` or `libhdesuoc.so`) in the `SHLIB_PATH` environment variable (`LD_LIBRARY_PATH` environment variable for the Solaris or Linux

Datareplicator, and `LIBPATH` environment variable for the AIX Datareplicator).

If the `SHLIB_PATH` environment variable (`LD_LIBRARY_PATH` environment variable for the Solaris or Linux Datareplicator, and `LIBPATH` environment variable for the AIX Datareplicator) is not specified or there is no shared library file at the directory path specified in the environment variable, Datareplicator executes import processing without using the send data UOC routine.

(1) Limitations on signals

- Do not conduct signal operations in a send data UOC routine's function.
- Do not use a function that controls processes (such as `exit()` or `abort()`).
- Do not execute `free()` at the address of the pre-update or post-update column data in `HDE_EXT_DATAINFO` within a send data UOC routine.

(2) Limitations on file manipulations

You can manipulate only user-specific files (that are open) within a send data UOC routine's function. The following are the file-related rules:

- Do not use the standard input, standard output, or standard error output.
- Do not read, write, or close any file that is not opened by the UOC routine.
- Do not manipulate any file that belongs to HiRDB or Datareplicator.
- Do not process the data that is passed to the send data UOC routine.
- You cannot inherit resources allocated within a send data UOC routine from one session to another (that is, the resources are not still available the next time the send data UOC routine is called).

(3) Limitations on shared memory

Do not use the same shared memory that is used by Datareplicator or by any other system.

(4) Limitations on data types

Columns of the following data types are not supported by send data UOC routines:

- Abstract data types (`SGMLTEXT`, `FREWORD`, `XML`)
- `BLOB` and `BINARY` types on which addition or update operations have been performed
- `BLOB` and `BINARY` types on which backward deletion updating has been performed.

(5) Limitations on SQL execution

Do not execute any SQL statement within or as an extension of the send data UOC routine's function. If executed, transaction management by Datareplicator might

become invalid.

(6) *Other limitations*

- Even if EBCDIK is specified as the transmission character code set, Datareplicator passes data in the character code set used before the EBCDIK conversion.
- When you create a send data UOC routine, do not overlook the fact that processing for the same target might be divided into multiple processes.
- The table ID and column IDs passed in the arguments are those in effect at the time of execution of the `hdeprep` command. If the UOC uses these IDs and a table is re-created after execution of the `hdeprep` command, you must correct the UOC routine.

8.3.6 Sample send data UOC routine

This section explains the sample UOC routine that is stored in Datareplicator.

(1) *Storage directory and filename*

Directory

UNIX Datareplicator: `/opt/hirdbds/lib/sample/`

Windows Datareplicator: `Datareplicator-installation-directory\sample\`

Filename

`hde_uocsample.c`

(2) *Contents of the source file*

The sample send data UOC routine is created on the basis of the following specifications:

- Check the stock table's product code and stock quantity and send the corresponding data.
- The conditional expressions are shown below. If the conditions are satisfied, the UOC routine sets `HDE_EXT_SEND` as its return value.
Product code = 10 and stock quantity is at least 100
or
Product code = 20 and stock quantity is at least 50
- Data in extraction tables other than the stock table are all subject to transmission processing unconditionally.

(3) *Sample library creation procedure*

The following example creates a sample library for HP-UX:

8. User Own Coding Routines

```
$ cc -c +z -I/opt/hirdbds/include hde_uocsample.c  
$ ld -b -o libhdesuoc.sl hde_uocsample.o
```

Chapter

9. Error Handling Procedures

This chapter explains the procedures for handling errors that occur during operation of source and target Datareplicators.

- 9.1 Error handling procedures for the source Datareplicator
- 9.2 Error handling procedures for the target Datareplicator
- 9.3 Error recovery method selection criteria
- 9.4 Initialization procedure during error recovery
- 9.5 Data linkage recovery via the system log file
- 9.6 Data linkage recovery via unload log files
- 9.7 Facility for recovering the extraction information queue file
- 9.8 Acquisition of untransmitted information due to import errors (update-SQL output facility)

9.1 Error handling procedures for the source Datareplicator

This section explains the error handling procedures for the source Datareplicator. For details about how to handle target system errors, see *9.2 Error handling procedures for the target Datareplicator* for HiRDB systems and see the applicable manuals for non-HiRDB systems.

9.1.1 Error handling procedures

This subsection explains the error handling procedures for the source Datareplicator.

(1) *Handling procedure*

To handle an error:

1. Check the error message.
Messages are output to the syslog file (event log in Windows). Check the messages.
2. Check the error information files listed below to see if they contain the same error messages that have been output to the syslog file (event log in Windows):
 - Extraction master error information file
 - Extraction node master error information file

A message (KFRB00051-I or KFRB00052-I) is output when an error information file is swapped or closed during Datareplicator operation. If this message is used as the basis for making a backup copy, first check its contents.

3. Use the `hdstrcredit` command to edit the following files, analyze them, and then save the results:
 - Extraction master trace file
 - Extraction node master trace file

4. Take action that is appropriate to the message.

For details about how to handle errors, see *9.1.2 Error handling methods*.

5. Take appropriate action after correcting the error.

Depending on the cause of the error, you might need to take some action that is appropriate to the process that resulted in the error. For details about the appropriate actions for each process that results in an error, see *9.1.3 Actions after correcting an error*.

Depending on the error, the source Datareplicator's process or shared memory (memory mapped file in Windows) might remain. In the UNIX edition of

Datareplicator, any remaining processes or shared memory will be deleted when the Datareplicator is restarted.

In the Windows edition of Datareplicator, use the following procedure to delete any remaining processes or memory mapped file:

Step	Description
1	Start the Task Manager, and on the Processes page obtain the ID of the Datareplicator process.
2	Terminate the process with the process ID obtained in step 1: <code>pdkill process-ID#</code>
3	Delete the memory mapped file under <code>installation-directory\temp\hde\spool\shm</code> .

#

If you are using Datareplicator Extension, use the `hdskill` command instead of the `pdkill` command.

(2) Contacting the system administrator

If you need to contact the system administrator about an error:

1. Obtain the following data:

- Messages output to the syslog file (event log in Windows)
- Source Datareplicator's status information
Execute the `hdestate` command to obtain status information.
- Core file (entire process's dump file)

This file is output in the directory specified in the `HDEPATH` environment variable in the following format:

`core process-ID`

The process ID is used to identify the process resulting in an error.

Note that, if the user who started the source Datareplicator does not have write permission for the directory specified in the `HDEPATH` environment variable, however, the core file is not created.

- Memory mapped file (Windows edition)
- Extraction master status file
- Extraction server status file
- Extraction master error information files
- Extraction node master error information files

- Extraction master trace files
 - Extraction node master trace file
 - Data linkage file
2. Check the nature of the error and take appropriate action.

In the event of an internal conflict error, contact the customer support center.

9.1.2 Error handling methods

The table below shows the possible causes of errors in the source system and the handling methods for such errors. If an inconsistency has occurred between the source and target databases due to an error, the source and target Datareplicators must be initialized. For details about the initialization procedure, see *9.4 Initialization procedure during error recovery*.

Table 9-1: Causes of errors and handling methods

Cause of error		Error handling method
Memory shortage (local memory)		<ul style="list-style-type: none"> • Terminate any unneeded processes. • Increase the memory size.
Memory shortage (shared memory)		<ul style="list-style-type: none"> • Terminate any unneeded processes that use the shared memory. • Increase the memory size.
Shortage of file descriptors		<ul style="list-style-type: none"> • Close any unneeded files. • Modify the system parameters (maximum number of file descriptors).
Socket establishment failure		<ul style="list-style-type: none"> • Eliminate the cause of the error on the basis of the detail code from the system call.
Port check error		Eliminate the cause of the error on the basis of the detail code.
Communication line failure		Reconnect the communication line.
Insufficient file space	Extraction information queue file	Send all update information from the extraction information queue file to the target system, and then terminate the source Datareplicator with the <code>hdestop</code> command. Increase the number of files or the file size, initialize the files with the <code>hdestart -i</code> command, then restart Datareplicator with the <code>hdestart</code> command. For details, take the appropriate action from the flowchart <i>Handling procedure when the extraction information queue file is full</i> .
	Status file during extraction processing	Send all update information from the extraction information queue file to the target system, and then terminate the source Datareplicator with the <code>hdestop</code> command. Increase the file size, initialize the file with the <code>hdestart -i</code> command, then restart Datareplicator with the <code>hdestart</code> command.

Cause of error		Error handling method
Invalid medium	Extraction information queue file	<p>If an error occurred in the extraction information queue file only: Eliminate the cause of the error, and then recover the extraction information queue file by using the facility for recovering the extraction information queue file. For details about the facility for recovering the extraction information queue file, see <i>9.7 Facility for recovering the extraction information queue file</i>.</p> <p>If an error occurred in another file in addition to the extraction information queue file: Eliminate the cause of the error, synchronize the data linkage environments at the source and target, initialize them, then re-create the target database on the basis of the source database.</p>
	Status file during extraction processing	Eliminate the cause of the error, synchronize the data linkage environments at the source and target, initialize them, then re-create the target database on the basis of the source database.
	Data linkage file	
	System log file [#]	
	Source database	Use the database recovery utility (<code>pdrrstr</code>) to restore the database from its backup copy and the log acquired since the backup was made. If you use any other method to restore the database, synchronize the data linkage environments at the source and target, initialize them, then re-create the target database on the basis of the source database.
Extraction information queue file full		<p>Use the <code>hdestate</code> command to check the status of the transmission process. If it is not running normally, eliminate the cause of the error, then restart the transmission process.</p> <p>To cancel data linkage, such as when you cannot correct the transmission process error, execute forced termination of the source Datareplicator. Then, synchronize the data linkage environments at the source and target, initialize them, and re-create the target database on the basis of the source database</p> <p>For details, take the appropriate action from the flowchart <i>Handling procedure when the extraction information queue file is full</i>.</p> <p>For details about how to handle errors caused by destinations that have become subject to reduced operation because the extraction information queue file became full, see <i>9.1.4 Procedures for handling errors at the target system</i>.</p>
Invalid update information		Save the status file and extraction information queue file obtained immediately after the invalid update information error occurred, then contact the customer support center. Eliminate the cause of the error, synchronize the data linkage environments at the source and target, initialize them, then re-create the target database on the basis of the source database
Invalid definition information		Correct the definitions for the source system or the source Datareplicator.

Cause of error	Error handling method
SQL error	Eliminate the cause of the error on the basis of the SQLCODE.
Internal conflict	Contact the customer support center.
Machine power interrupt	Restart the machine.
Bus error	Contact the customer support center.
Invalid signal reception (<i>sigkill</i>)	UNIX edition of Datareplicator: If the source Datareplicator received an invalid signal, execute the <code>hdeshmclean</code> command and restart the machine. If the target Datareplicator received an invalid signal, execute the <code>hdsshmclean</code> command and restart the machine. Windows edition of Datareplicator: Restart the machine.

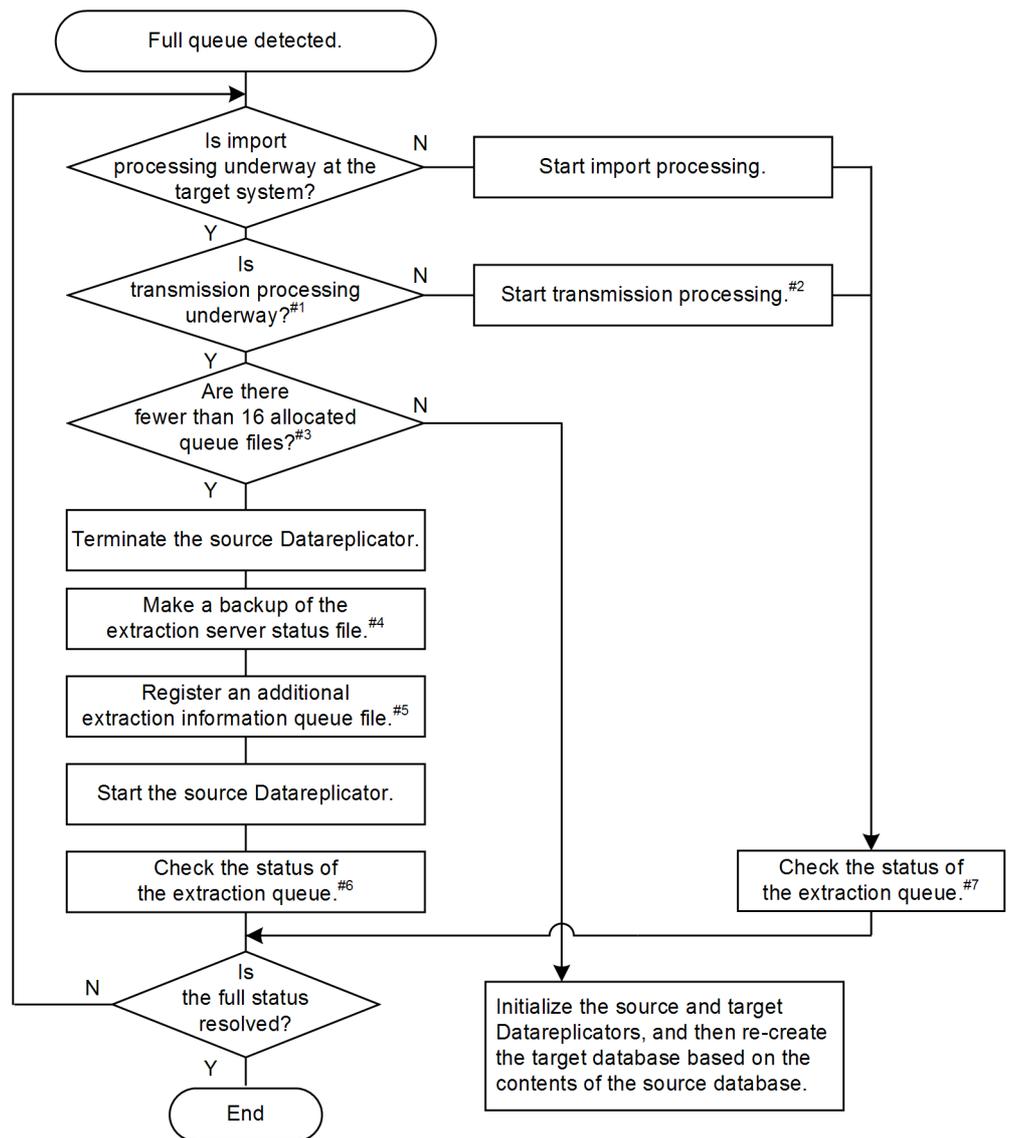
#: If you are using dual system log files, this applies when errors occur in both editions.

Handling procedure when the extraction information queue file is full:

An error message is displayed when the extraction information queue file becomes full. The message that is displayed depends on the process that issues the message, as follows:

- Extraction node master: KFRB00042-E
- Extraction master: KFRB00701-E

The following figure shows the handling procedure when these messages are displayed:



#1

Execute the `hdestate` command to determine whether transmission processing is underway.

#2

If necessary, eliminate the cause of the termination and use the `hdestart -s`

command to start transmission processing.

#3

From the execution results of the `hdestate` command, determine how many extraction information queue files have been allocated.

#4

Make a backup in case a registration error occurs in the additional extraction information queue file. For details about the backup method, see *6.4.2(7)(b) Backing up the extraction server status file*.

#5

You can register additional extraction information queue files with the `hdemodq` command. For details about the `hdemodq` command, see *6.4.2 (6) (b) Command for changing the organization of extraction information queue files (hdemodq command)* or the `hdemodq` command in Chapter 7. *Command Syntax*.

#6

After the time needed to read the entire extraction information queue files has elapsed (about 1 second per megabyte), execute the `hdestate` command at each transmission interval (1 minute if the specified value is 0) to check the processing status on the extraction information queue files. If `Queue read position` indicates the extraction information queue file that follows `Queue write position`, the full status has not been resolved, in which case add an extraction information queue file.

#7

After the transmission interval has elapsed (1 minute if the specified value is 0), execute the `hdestate` command at each transmission interval to check the processing status on the extraction information queue. If `Queue read position` indicates the extraction information queue file that follows `Queue write position`, the full status has not been resolved, in which case add an extraction information queue file.

9.1.3 Actions after correcting an error

The following table shows the source Datareplicator processing for each process resulting in an error and the action to be taken by the user after correcting the error.

Table 9-2: Actions after correcting an error

Process resulting in an error	Source Datareplicator processing	User's action
Extraction master process	Terminates the source Datareplicator (including the source Datareplicator at each back-end server). Outputs the error message to the extraction master error information file and extraction node master error information file.	After correcting the error, restart the source Datareplicator with the <code>hdestart</code> command.
Extraction node master process	Terminates all extraction and transmission processing at the server machine resulting in the error. Outputs the error message to the extraction node master error information file.	After correcting the error, use the <code>hdestart</code> command to restart the extraction and transmission processing at the back-end server where the error occurred.
Extraction process	Terminates extraction processing at only the back-end server where the error occurred. Outputs the error message to the extraction node master error information file.	After correcting the error, use the <code>hdestart -e</code> command to restart the extraction processing that was terminated.
Transmission process	Terminate only the transmission processing that resulted in the error. Outputs the error message to the extraction node master error information file.	After correcting the error, use the <code>hdestart -s</code> command to restart the transmission processing that was terminated.
Transmission master process	Terminates all transmission processing. Outputs the error message to the extraction node master error information file.	After correcting the error, use the <code>hdestart -s</code> command to restart the transmission processing that was terminated.
Extraction command process	After issuing the message, cancels command execution. Outputs the error message to the standard output.	After correcting the error, re-execute the command.
Activity trace collection process	In the event of an error, issues a message and stops collecting activity trace information. However, Datareplicator processing continues.	Check the message and eliminate the cause of the error.

9.1.4 Procedures for handling errors at the target system

This section explains the procedures for handling errors at the target system for the following situations:

- When conformity between the source and target databases is lost
- Re-creating the target database when transmitting to multiple target systems

- When some destinations have become subject to reduced operation because the extraction information queue file became full

(1) When conformity between the source and target databases is lost

If conformity between the source and target databases is lost because of an error at the target system, you must re-create the target database. Synchronize the data linkage environments at the source and target, initialize them, then re-create the target database on the basis of the source database. For details about how to re-create a target database, see 6.5 *Handling of the source HiRDB*.

(2) Re-creating the target database when transmitting to multiple target systems

If an error occurs at a target system while executing transmission from one source system to multiple target systems, you can initialize transmission processing for only the erroneous target system by using the partial initial start method at the source system. In this case, execute partial initial start only for the transmission processing to the erroneous target system, and then re-create the target database at that target system.

To re-create a target database when transmitting to multiple target systems:

1. Terminate the applications at the source HiRDB to maintain conformity between the source and target databases that are subject to the data linkage.
2. Use the `pdhold` command to shut down the source database and suppress updating of the databases subject to the data linkage.
3. Terminate the source Datareplicator with the `hdestop` command.
4. Terminate the target Datareplicator with the `hdsstop` command.
5. Re-create the target database on the basis of the source database only at the erroneous target system. In this case, the HiRDB Dataextractor enables you to efficiently re-create the target database from the source database.
6. Specify the destination resulting in the error and execute partial initial start on the source Datareplicator (with the `hdestart -i -S` command).
7. Initialize the target Datareplicator's environment at the target system resulting in the error (with the `hdsstart -i` command).
8. Use the `hdestart` command to start the source Datareplicator.
9. Use the `pdrels` command to release the source database that is subject to data linkage from shutdown status.
10. Restart the applications at the source HiRDB.

(3) When some destinations have become subject to reduced operation because the extraction information queue file became full

If some destinations have become subject to reduced operation because the extraction information queue file became full, you must re-create the target databases at the target

systems that have been placed in reduced operation mode. The handling procedure is the same as when an error occurs at a target system while executing transmission to multiple target systems. For details, see (2) above.

9.1.5 User own coding routine error handling procedure

If an error occurs during operation of a UOC routine, the source Datareplicator takes action on the basis of the UOC routine's return value. For details about UOC routine return values, see Chapter 8. *User Own Coding Routines*.

If a UOC routine terminates abnormally, the target Datareplicator outputs a message and terminates abnormally.

9.1.6 Handling of file errors during file-duplexed operation

If a file error occurs while file-duplexed operation is underway, file recovery is required. This subsection describes how to handle such errors.

(1) *When an error occurs in both primary and secondary files*

If an error occurs in both the files (primary and secondary files), initialize Datareplicator.

(2) *When an error occurs in either primary or secondary file*

If an error occurs in only one of the files (either the primary file or the secondary file), the error handling method depends on the file resulting in the error.

Files that are recovered by Datareplicator automatically

Datareplicator recovers the files listed below automatically when an error occurs in any of these files. This recovery takes place when the process that uses the file resulting in the error is started.

- Extraction master status file
- Extraction server status file
- Data linkage file
- Import master status file
- Import status file

Files that must be recovered manually

If an error occurs in either of the files listed below, you must use the current file copy command (*hdefcopy* or *hdsfcopy*) to recover the erroneous file. For details about the command, see the *hdefcopy* or *hdsfcopy* command in Chapter 7. *Command Syntax*.

- Extraction information queue file
- Import information queue file

9.2 Error handling procedures for the target Datareplicator

This section explains the error handling procedures for the target Datareplicator. For details about how to handle source system errors, see *9.1 Error handling procedures for the source Datareplicator* for HiRDB systems and see the applicable manuals for non-HiRDB systems.

9.2.1 Error handling procedures

This subsection explains the error handling procedures for the target Datareplicator.

(1) Handling procedure

To handle an error:

1. Check the error message.

The messages are output to the syslog file (event log in Windows). Check the messages.

2. Check the error logs in the import error information file to see if they contain the same error messages that have been output to the syslog file (event log in Windows).

A message (KFRB00051-I or KFRB00052-I) is output when an error information file is swapped or closed during Datareplicator operation. If this message is used as the basis for making a backup copy, first check its contents.

3. Use the `hdstrcredit` command to edit the import trace file, analyze the file, and then store the results.
4. Take action that is appropriate to the message.

For details about how to handle errors, see *9.2.2 Error handling methods*.

5. Take appropriate action after correcting the error.

Depending on the cause of the error, you might need to take some action that is appropriate to the process that resulted in the error. For details about the appropriate actions for each process that results in an error, see *9.2.3 Actions after correcting an error*.

Depending on the error, the target Datareplicator's process or shared memory (memory mapped file in Windows) might remain. In the UNIX edition of Datareplicator, any remaining processes or shared memory will be deleted when the Datareplicator is restarted.

In the Windows edition of Datareplicator, use the following procedure to delete any remaining processes or memory mapped file.

Step	Description
1	Start the Task Manager, and on the Processes page obtain the ID of the Datareplicator process.
2	Terminate the process with the process ID obtained in step 1. <code>pdkill process-ID#</code>
3	Delete the memory mapped file under <code>installation-directory\temp\hde\spool\shm</code> .

#

If you are using Datareplicator Extension, use the `hdskill` command instead of the `pdkill` command.

(2) Contacting the system administrator

If you need to contact the system administrator about an error:

1. Obtain the following data:

- Messages output to the syslog file (event log in Windows)
- Import Datareplicator's status information
Execute the `hdsstate` command to obtain status information.

- Binary files indicating the error information

The following two files are output in the directory specified in the `HDSPATH` environment variable:

- `TCPDMP`
- `rfc_core`

Note that if the user who started the target Datareplicator does not have write permission for the directory specified in the `HDSPATH` environment variable, the `rfc_core` file is not created.

- Memory mapped file (Windows edition)
- Import status files
- Import master status file
- Import error information files
- Unimported information files
- Import trace files
- Definition files (if they might be changed)

2. Check the nature of the error and take appropriate action.

In the event of an internal conflict error, contact the customer support center.

9.2.2 Error handling methods

The table below shows the possible causes of errors in the target system and the handling methods for such errors. If an inconsistency has occurred between the source and target databases due to an error, the source and target Datarepliators must be initialized. For details about the initialization procedure, see *9.4 Initialization procedure during error recovery*.

Table 9-3: Causes of errors and handling methods

Cause of error		Error handling method
Memory shortage (local memory)		<ul style="list-style-type: none"> • Terminate any unneeded processes. • Increase the memory size.
Memory shortage (shared memory)		<ul style="list-style-type: none"> • Modify the <code>defshmsize</code> operand value in the import environment definition. • Terminate any unneeded processes that use the shared memory. • Increase the memory size.
Shortage of file descriptors		<ul style="list-style-type: none"> • Close any unneeded files. • Modify the system parameters (maximum number of file descriptors).
Socket establishment failure		Eliminate the cause of the error on the basis of the detail code from the system call.
Port check error		Eliminate the cause of the error on the basis of the detail code.
Communication line failure		Reconnect the communication line and restart the source system.
Data reception sequence error		Contact the source system's customer engineer.
Insufficient file space	Import information queue file	Use the <code>hdsstate</code> command to see if all the update information in the import information queue file has been imported (all update information has been imported when the write position of <code>COMMUNICATION INFORMATION</code> is the same as the read position of <code>REFLECTION INFORMATION</code>). If all update information has been imported but the error status indicated by <code>KFRB00005-E</code> (insufficient queue file space) still remains, execute the <code>hdsstop -t immediate</code> or <code>force</code> command to terminate the target Datarepliator. Then increase the number of files or the file size and restart Datarepliator with the <code>hdsstart -i</code> command. The source Datarepliator automatically re-executes transmission processing; there is no need to take any special action.
	Import status file	After importing all update information in the import information queue file, terminate the target Datarepliator with the <code>hdsstop</code> command. Then increase the file size and restart Datarepliator with the <code>hdsstart -i</code> command.
Invalid update information		Contact the customer support center.

Cause of error		Error handling method
Invalid definition information		Correct the definitions for the source system or the target Datareplicator.
Invalid medium	Import information queue file	After correcting the error, synchronize the data linkage environments at the source and target, initialize them, then re-create the target database on the basis of the source database.
	Import status file	
	Import master status file	
SQL error		Eliminate the cause of the error on the basis of the SQLCODE.
Error during update information input command processing		<p>When an error occurs during extraction processing Processing continues, because there is no effect on the update information input processing. You can restart extraction processing with the startup command (<code>hdestart</code>).</p> <p>When an error occurs during the current import processing Processing continues, because there is no effect on the update information input processing. You can restart the current import processing with the startup command (<code>hdsrfectl</code>).</p> <p>When an error occurs during update information input processing Import processing continues to input update information. To restart the update information input processing, execute the <code>hdssamqin</code> command with the <code>-o</code> option specified. The <code>-o</code> option is applicable only in the event of an error. There is no need to execute this command in the case of abnormal termination that allows message output.</p> <p>When an error occurs during import processing for update information input processing Update information input processing continues. You can restart import processing with the startup command (<code>hdsrfectl</code>).</p>
Internal conflict		Contact the customer support center.
Machine power interrupt		Restart the machine.
Bus error		Contact the customer support center.
Invalid signal reception (<code>sigkill</code>)		<p>UNIX edition of Datareplicator: If the source Datareplicator received an invalid signal, execute the <code>hdeshmclean</code> command and restart the machine. If the target Datareplicator received an invalid signal, execute the <code>hdsshmclean</code> command and restart the machine.</p> <p>Windows edition of Datareplicator: Restart the machine.</p>

Checking the transaction status when using the two-phase commit method for

such errors

If the two-phase commit method is being used for synchronization point processing and an SQL error occurs while executing a synchronization point processing request, the status of the transaction that is being executed by Datareplicator's import facility is displayed as `XID`, which is displayed by executing HiRDB's `pdls -d trn -a` command. `XID` displays information about the transaction in the following format:

```
XID=aaaaaaaaabbccddd,eeeeeeffgghhhh
```

aaaaaaaa: Name of the executing process. If it is a UOC routine, `hdsuocg` is displayed; otherwise, `hdssqle` is displayed.

bb: Datareplicator identifier (00-ff)

cc: Data linkage identifier (00-ff)

dddd: 0000

eeeeeee: Name of the import group

ff: Import group sequence number (sequential import group number assigned for each data linkage identifier)

gg: SQL process number (sequential SQL process number assigned for each import group)

hhhh: 0000

Transaction settlement when using the two-phase commit method

When synchronization point processing is performed using the two-phase commit method, do not use a command to settle a transaction except when canceling data linkage, and then executing an initial start. Normally, the target Datareplicator's transaction is settled automatically when the target Datareplicator starts. If the target Datareplicator terminates due to abnormal termination of the target HiRDB, a transaction might remain unsettled. In this case, restart the target Datareplicator with the procedure shown below.

To restart the target Datareplicator:

1. Execute forced termination of the target Datareplicator with the `hdsstop -t force` command.
2. Eliminate the cause of the abnormal termination and restart the target HiRDB.
3. Execute the `pdls -d trn -a` command to obtain a list of unsettled transactions, and settle them on the basis of the displayed information. Then restart the target Datareplicator.

The following table shows the transaction information that is displayed and the action to be taken by the user:

Displayed information			User's action
PROGRAM	B-SVID	STATUS	
Other than hdssqle	N/A	N/A	Because this is not a Datareplicator transaction, settle it on the basis of instructions provided by the program that generated the transaction.
hdssqle	**...*	N/A	No user action is necessary, because all transactions whose TRNGID is shown as **...* are settled automatically when the target Datareplicator starts.
	Other than **...*	FORGETTING(?,w) ?: Any character	Use the <code>pdfgt</code> command to settle a transaction with this TRNGID.
		Other than FORGETTING(?,w)	Compare the XID of the transaction with this TRNGID with the XID in the KFRB3072-E message output to the error information file, then take one of the following actions: There is a KFRB3072-E message with the same XID: Use the <code>pdcmr</code> command to settle the transaction. There is no KFRB3072-E with the same XID: Use the <code>pdrbk</code> command to settle the transaction.

9.2.3 Actions after correcting an error

The following table shows the target Datareplicator processing for each process resulting in an error and the action to be taken by the user after correcting the error.

Table 9-4: Actions after correcting an error

Process resulting in an error	Target Datareplicator processing	User's action
Import master process	Terminates the target Datareplicator.	After correcting the error, restart the target Datareplicator with the <code>hdsstart</code> command.
Import communication master process Reception process	Stops communication and reception processing. When the source system issues a reconnection request, the target Datareplicator automatically restarts the communication and reception processing. Resumes import processing.	N/A
	Stops data reception and does not recover the communication or reception processing. Resumes import processing.	Terminate the target Datareplicator with the <code>hdsstop</code> command. Then eliminate the cause of the error and restart the target Datareplicator with the <code>hdsstart</code> command. ^{#1}

Process resulting in an error	Target Datareplicator processing	User's action
Import definition server process Import process Import SQL process	Terminates the import processing that resulted in an error. Resumes reception processing.	After correcting the error, use the <code>hdsrcctl</code> command to restart the import processing that resulted in the error. ^{#2}
Activity trace collection process	Outputs a message and stops collecting activity trace information. Resumes Datareplicator processing.	Check the message and eliminate the cause of the error.
Update information input process	Terminates the <code>hdssamqin</code> command processing.	Re-execute the <code>hdssamqin</code> command. For details about the command options, see the <i>hdssamqin</i> command in Chapter 7. <i>Command Syntax</i> .

#1: If conformity between the source and target databases has been lost, synchronize the data linkage environments at the source and target, initialize them, then re-create the target database on the basis of the source database.

#2: If conformity between the source and target databases has been lost, terminate the target Datareplicator with the `hdstop` command. Then synchronize the data linkage environments at the source and target, initialize them, and re-create the target database on the basis of the source database.

9.2.4 Procedures for handling errors at the source system

This section explains the procedures for handling errors at the source system for the following situations:

- When conformity between the source and target databases is lost
- Re-creating the target database when importing from multiple source systems

(1) *When conformity between the source and target databases is lost*

If conformity between the source and target databases is lost because of an error at the source system, you must re-create the target database. Synchronize the data linkage environments at the source and target, initialize them, then re-create the target database on the basis of the source database.

(2) *Re-creating the target database when importing from multiple source systems*

If an error occurs at a source system while importing update information from multiple source systems into a single target system, you can initialize the import processing for only the erroneous source system by using partial initial start at the target system. In this case, execute partial initial start for the import processing from the erroneous source system only, and then re-create the target database corresponding to the

erroneous source system.

To re-create the target database when importing from multiple source systems:

1. Shut down the database subject to data linkage at the source system where the error occurred, then suppress updating of the database.
2. If the source system is a HiRDB, terminate the source Datareplicator with the `hdestop` command. If the source system is a mainframe, terminate the source XDM/DS.
3. Terminate the target Datareplicator with the `hdsstop` command.
4. Re-create only the target database corresponding to the erroneous source system on the basis of the source database. In this case, the HiRDB Dataextractor enables you to efficiently create the target database from the source database.
5. Initialize the data linkage environment at the source system.
 If the source system is a HiRDB, initialize the source Datareplicator's environment (with the `hdestart -i` command).
 If the source system is a mainframe, initialize the source XDM/DS's environment.
6. Execute partial initial start of the target Datareplicator specifying the erroneous source system (with the `hdsstart -i -D` command).
7. If the source system is a HiRDB, start the source Datareplicator with the `hdestart` command. If the source system is a mainframe, start the source XDM/DS.
8. Release from shutdown status the source database that is subject to data linkage.

9.2.5 User own coding routine error handling procedure

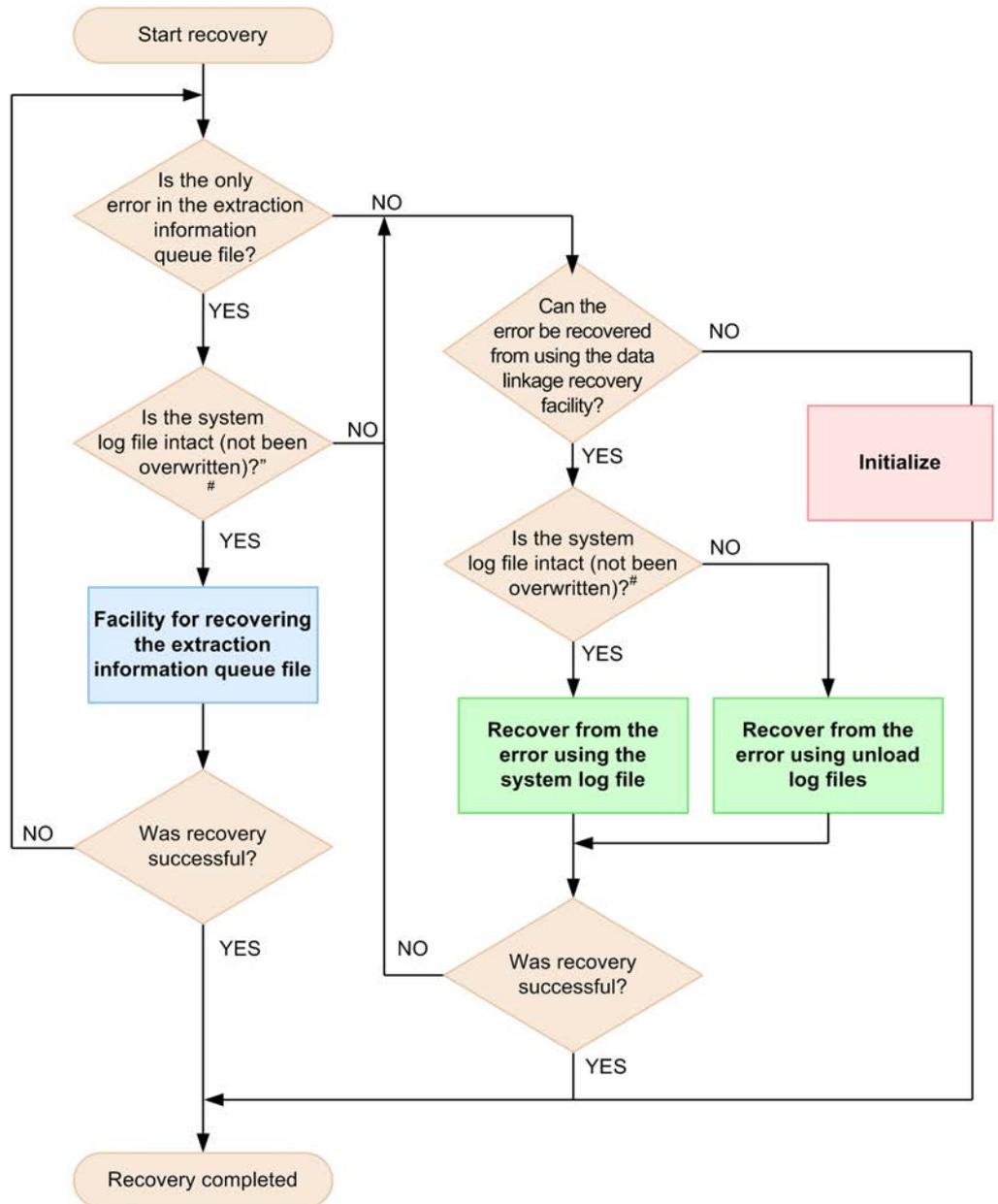
If an error occurs during operation of a UOC routine, the target Datareplicator takes action on the basis of the UOC routine's return value. For details about UOC routine return values, see Chapter 8. *User Own Coding Routines*.

If a UOC routine terminates abnormally, the target Datareplicator outputs a message and terminates abnormally.

9.3 Error recovery method selection criteria

The available recovery methods depend on the situation in which an error occurred. The following figure shows the criteria for selecting the appropriate error recovery method.

Figure 9-1: Recovery method selection criteria



#

If you cannot determine whether the system log file has been overwritten, choose

the YES branch and continue with recovery processing. If an error occurs during recovery processing because of an overwritten system log file, re-execute the recovery processing starting from the NO branch.

If an error occurred only in the extraction information queue file, use the facility for recovering the extraction information queue file. If you use the data linkage recovery facility, you must recover the error at both the source and the target. If you use the facility for recovering the extraction information queue file, you need to recover the error only at the source.

The following table lists the errors that can be recovered by using the data linkage recovery facility (source system).

Table 9-5: Errors that can be recovered by using the data linkage recovery facility (source system)

Error event		Data linkage recovery facility		
		Recovery via the system log file	Recovery via unload log files	
Command error	The source Datareplicator was initialized.	Y	NR	
	The <code>pdrplstop</code> command was executed (to stop data linkage) or the <code>hdestart -i</code> command was executed (to perform an initial start).	Y	NR	
Disk error	Extraction information queue file	The source HiRDB's system log file has not been overwritten.	NR [#]	
		The source HiRDB's system log file has been overwritten.	N	
	Extraction server status file		Y	NR
	Data linkage file		Y	NR
Insufficient disk capacity	Extraction information queue file	Y	NR	

Legend:

Y: Can be recovered.

NR: Can be recovered, but this recovery method is not recommended.

N: Cannot be recovered.

#

Use the facility for recovering the extraction information queue file.

The following table lists the errors that can be recovered by using the data linkage recovery facility (target system).

Table 9-6: Errors that can be recovered by using the data linkage recovery facility (target system)

Error event		Data linkage recovery facility	
		Recovery via the system log file	Recovery via unload log files
Command error	The target Datareplicator was initialized.	Y	NR
Disk error	Import information queue file	Y	NR
Insufficient disk capacity	Import status file	Y	NR

Legend:

Y: Can be recovered.

NR: Can be recovered, but the recovery method indicated by Y is recommended.

For details about initialization, see *9.4 Initialization procedure during error recovery*.

For details about recovery using the system log file with the data linkage recovery facility, see *9.5 Data linkage recovery via the system log file*. For details about recovery using unload log files, see *9.6 Data linkage recovery via unload log files*.

For details about the facility for recovering the extraction information queue file, see *9.7 Facility for recovering the extraction information queue file*.

9.4 Initialization procedure during error recovery

If an inconsistency occurs between the source and target databases due to an error, the source and target Datareplicators must both be initialized. This section explains how to initialize the Datareplicators when an inconsistency occurs between the source and target databases.

9.4.1 Errors that require initialization of Datareplicators

Listed below are errors that result in an inconsistency between the source and target databases. If any of the following errors has occurred, initialize the source and target Datareplicators.

- Corruption of the following files (such as disk corruption and data deletion):

Source Datareplicator's files:

- Extraction master status file
- Extraction server status file
- Data linkage file
- Extraction information queue file
- Extraction definition preprocessing file
- Duplexing control file

Target Datareplicator's files:

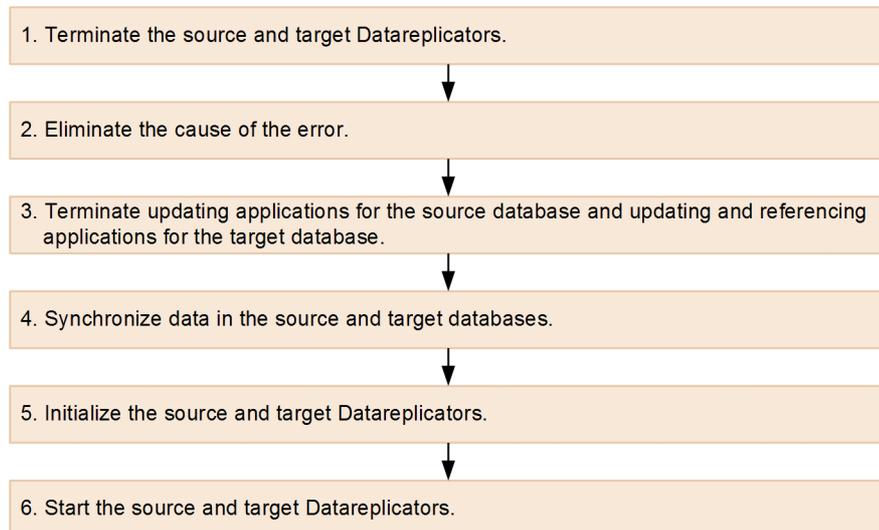
- Import master status file
 - Import status file
 - Import information queue file
 - Duplexing control file
- SQL errors due to an inconsistency between the source and target databases, such as data duplication and `Not Found` errors
 - Code conversion errors due to entry of invalid data
 - Attempt to import a data type that is not supported by the target Datareplicator as a column subject to extraction processing
 - Overwriting of a system log file that contains unextracted data
 - Termination of data linkage by the `pdrplstop` command
 - Re-creation or initialization of the target or source database (RDAREA, system log file)

9.4.2 Datareplicator initialization procedure during error recovery

This subsection explains how to initialize the Datareplicators if an inconsistency occurs between the source and target databases.

The following figure shows the procedure.

Figure 9-2: Initialization procedure during error recovery



Notes:

- Steps 1, 2, and 3 can be performed in any order.
- You must start the source and target databases to perform the steps 4 through 6.

The initialization procedure during error recovery that is shown in the figure is explained below. The numbers in the explanation below correspond to the numbers in the figure.

1. Terminate the source and target Datareplicators.

No.	Task	Execution command	Check item
1	Stop the source Datareplicator.	<code>hdestop</code>	--
2	Verify that the source Datareplicator has terminated.	<code>hdestate</code>	Verify that the <code>KFRB04411-E</code> message has been output to the standard error.
3	Terminate the target Datareplicator.	<code>hdsstop</code>	--

9. Error Handling Procedures

No.	Task	Execution command	Check item
4	Verify that the target Datareplicator has terminated.	hdsstate	Verify that the KFRB04302-E message has been output to the standard error.

Legend:

--: Not applicable

2. Eliminate the cause of the error.
3. Terminate updating applications for the source database and updating and referencing applications for the target database.
4. Synchronize data in the source and target databases.

Use a program such as HiRDB Datareplicator to copy data from the source tables to the target tables (initial data creation) in the batch mode to synchronize data in the source and target databases.

5. Initialize the source and target Datareplicators.

No.	Task	Execution command	Check item
1	Terminate data linkage with HiRDB.	pdrplstop -f	Verify that the KFPS05141-I message has been output to the standard output.
2	Initialize the source Datareplicator.	hdestart -i (Enter Y in the response message)	Verify that the KFRB00504-I message has been output to msterrfile1 or msterrfile2.
3	Create an extraction definition preprocessing file.	hdeprep -f <i>extraction-definition-file#</i>	Verify that the KFRB04500-I message has been output to the standard output.
4	Start data linkage with HiRDB.	pdrplstart	Verify that the KFPS05140-I message has been output to the standard output.
5	Initialize the target Datareplicator.	hdsstart -i -q#	Verify that the KFRB04216-I message has been output to the standard output (Event Viewer in Windows).

#

Specify options as necessary.

6. Start the source and target Datareplicators.

No.	Task	Execution command	Check item
1	Start the target Datareplicator.	hdsstart [#]	Verify that the KFRB00100-I message has been output to errfile1 or errfile2.
2	Start the source Datareplicator.	hdestart [#]	Verify that the KFRB00502-I message has been output to msterrfile1 or msterrfile2.

#

Specify options as necessary.

9.5 Data linkage recovery via the system log file

The data linkage recovery facility is applied in the event of termination of data linkage. It enables you to recover the integrity of data linkage by re-extracting unimported update information that was lost due to the error, recover the extraction information queue file based on the re-extracted information, and then send the data to the target.

The data linkage recovery facility is used to restore conformity of data linkage quickly in a large-sized system where it would normally take HiRDB Dataextractor several days to restore the entire table.

The data linkage recovery facility supports two methods:

- Recovery via the system log file
- Recovery via unload log files

This section explains data linkage recovery via the system log file. For details about data linkage recovery via unload log files, see *9.6 Data linkage recovery via unload log files*.

Note:

If used for normal operation by mistake, the data linkage recovery facility might result in inconsistent data linkage. Make sure that this facility is used only in the event of a data linkage error.

9.5.1 Overview of data linkage recovery via the system log file

Data linkage recovery via the system log file provides the following benefits:

- The recovery procedure is simple.
- The following files can be recovered without having to terminate applications at the source:
 - Extraction master status file
 - Data linkage file
 - Extraction server status file
 - Import master status file
 - Import information queue file
 - Import status file

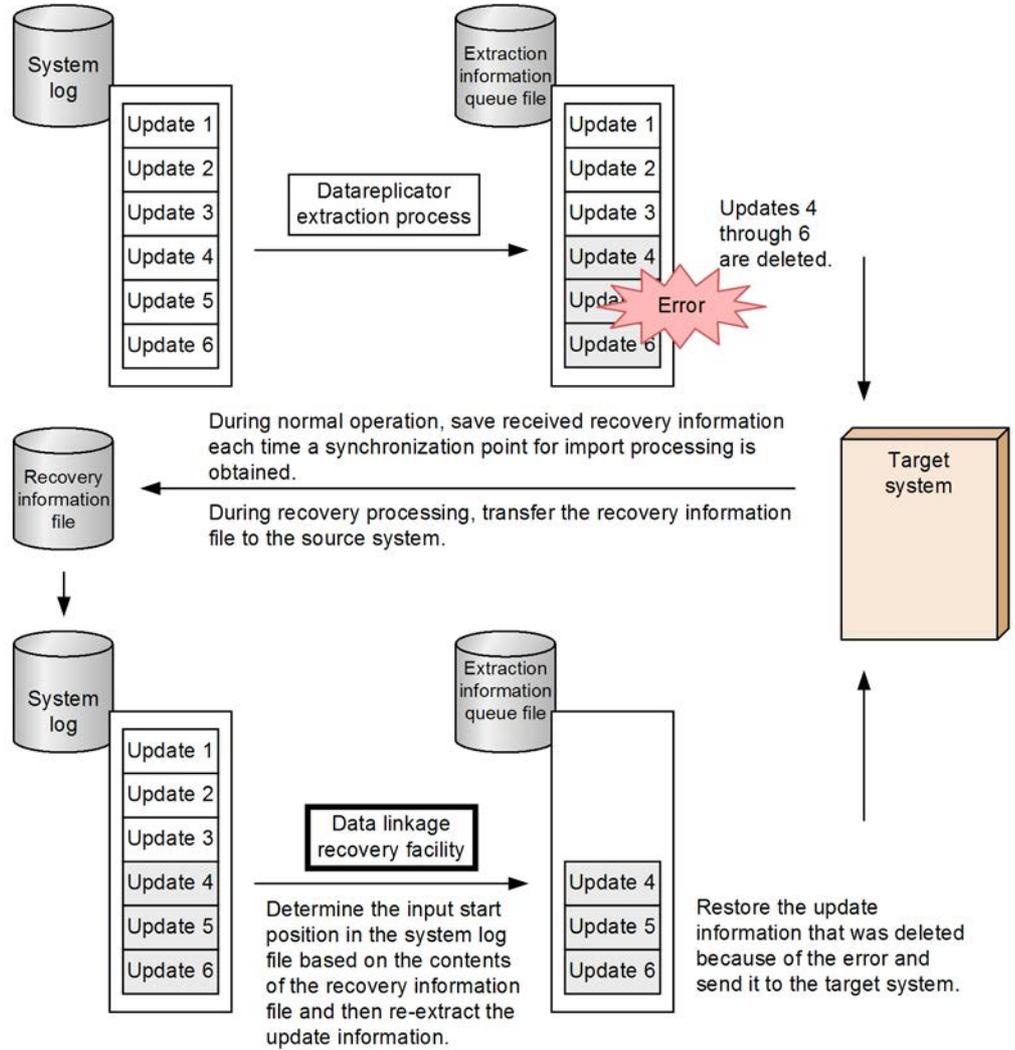
However, note the following:

- This recovery method cannot be used if the system log file has been overwritten.

- The amount of transmission data increases because during normal operation various recovery information is added to the normal update information for sending to the target Datareplicator.

The following figure provides an overview of data linkage recovery via the system log file in the event of an error in the extraction information queue file.

Figure 9-3: Overview of data linkage recovery via the system log file



9.5.2 Prerequisites for data linkage recovery via the system log file

(1) Combinations of supported versions and products

The following table shows the combinations of the versions and products that support data linkage recovery via the system log file.

Table 9-7: Combinations of the versions and products that support data linkage recovery via the system log file

Target system		Source system	
		Datareplicator	
		08-03 or earlier	08-04 or later
Datareplicator	08-03 or earlier	N	N
	08-04 or later	N	Y
XDM/DS		N	N

Legend:

Y: Supports data linkage recovery via the system log file.

N: Does not support data linkage recovery via the system log file.

(2) Prerequisites

The prerequisites for using data linkage recovery via the system log file are as follows:

- The source DBMS is HiRDB.
- HiRDB's system log file has not been overwritten.
- `true` is specified in the `recover_info_send` operand in the extraction system definition.
- A recovery information file is available.

(3) Preparing for data linkage recovery via the system log file

The preparations described in this subsection are required to perform data linkage recovery via the system log file.

(a) Creating a recovery information file

When you specify `true` in the `recover_info_send` operand in the extraction system definition, a recovery information file is created under the target Datareplicator's `$HDSPATH` directory. Recovery information will be stored in this file. This information is required to determine the input start position in the system log file during recovery processing. The following table describes the contents the recovery

information file.

Table 9-8: Contents of the recovery information file

Item	Contents	Remarks
File name	rcvrfile_xx_yy	xx: Target Datareplicator identifier (hdsid) yy: Data linkage identifier (dsidxx) Specify both as two hexadecimal characters. For alphabetical characters, use lower-case letters.
Location of file creation	Under the target Datareplicator's \$HDSPATH	--
Unit of creation	One file per target Datareplicator data linkage identifier (dsidxx)	--
File type	Regular file or character special file	If the target Datareplicator is in a system switchover environment, use a character special file.
File size	(1 + n) KB	The size depends on the import method: <ul style="list-style-type: none"> • Transaction-based import method n: 1 • Table-based import method n: Number of import groups Recovery information is stored for each import group.
Creation timing	When port check information is received	--
Update timing	When an import synchronization point is obtained (when the import status file is updated)	Update information is overwritten, not accumulated.

Legend:

--: Not applicable

Transmission of recovery information

Because the recovery information is sent to the target Datareplicator for each transaction that occurs during extraction processing, the amount of transmission data and the usage amount of the target Datareplicator's import information queue file increase during normal operation. The formula for determining the amount of increase in the resources needed to send recovery information is shown below. When you design the resources, take into account this increase.

- Amount of transmission data

$$(\uparrow TRUN_NUM / SND_INT \uparrow) \times 144 \text{ (bytes)}$$

- Size of the import information queue file

$(\uparrow TRN_NUM / SND_INT \uparrow) \times 144$ (bytes)
--

- Size of the extraction information queue file

$144 \times TRN_NUM$ (bytes)

TRN_NUM: Number of transactions that updated the table subject to extraction processing

SND_INT: *recover_info_send_interval* operand value (1 if omitted)

Recovery information transmission interval

To minimize the amount of increase in transmission data during normal operation, be sure to specify a recovery information transmission interval in the *recover_info_send_interval* operand in the extraction system definition.

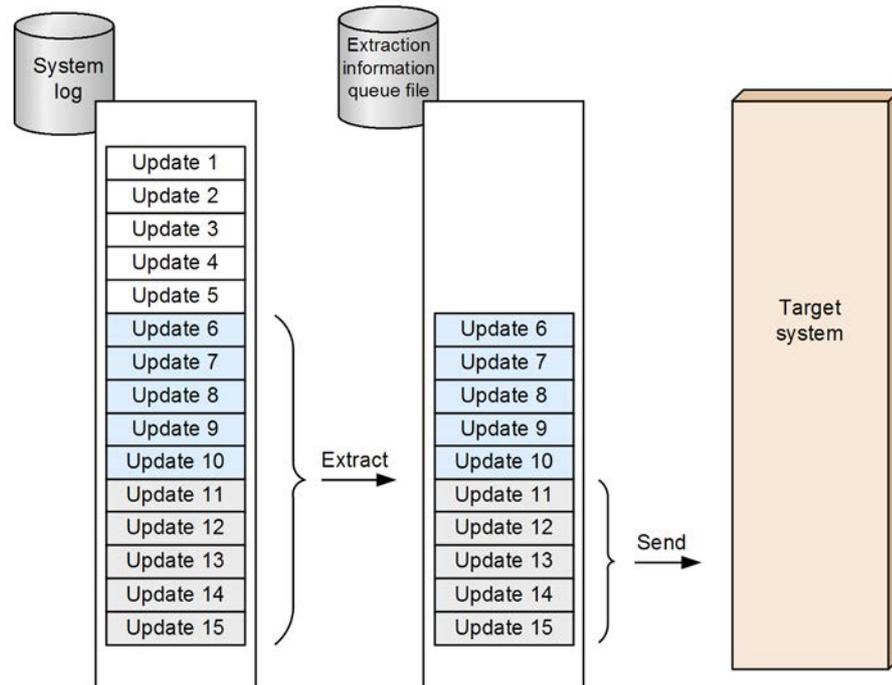
We recommend that the value you specify in this operand be based on the average number of transactions per transmission interval, in the range from 1 to 32,767, so that recovery information will be added at least once per transmission interval (*sendintvl* value specified in the transmission environment definition).

The following explains what happens if you specify a value other than 1 in the *recover_info_send_interval* operand:

After a transmission process starts during normal operation, Datareplicator adds recovery information to the transmission data for the first transaction to be sent. Thereafter, Datareplicator sends recovery information every specified number of transactions. In this case, during recovery processing, Datareplicator will input and extract update information including for instances that do not require recovery.

The following figure shows an example of recovery information transmission when 5 is specified in the *recover_info_send_interval* operand.

Figure 9-4: Example of recovery information transmission

**Explanation:**

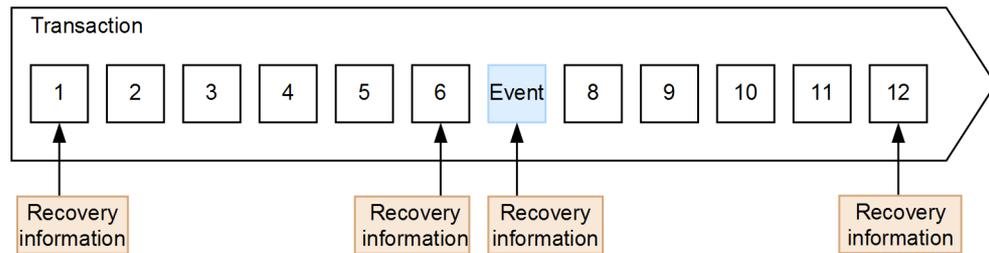
The update information that needs recovery due to an error is Update 11 through Update 5. Because the `recover_info_send_interval` operand is specified, Datareplicator inputs and extracts the update information starting at Update 6. The update information that is not needed for recovery will be input and extracted, but not sent.

Note:

Recovery processing cannot be performed, even if update information whose recovery is not needed has been overwritten.

Regardless of this operand's value, Datareplicator sends the recovery information during event transmission. The following figure shows an example of recovery information transmission during event transmission when 5 is specified in the `recover_info_send_interval` operand.

Figure 9-5: Example of recovery information transmission during event transmission



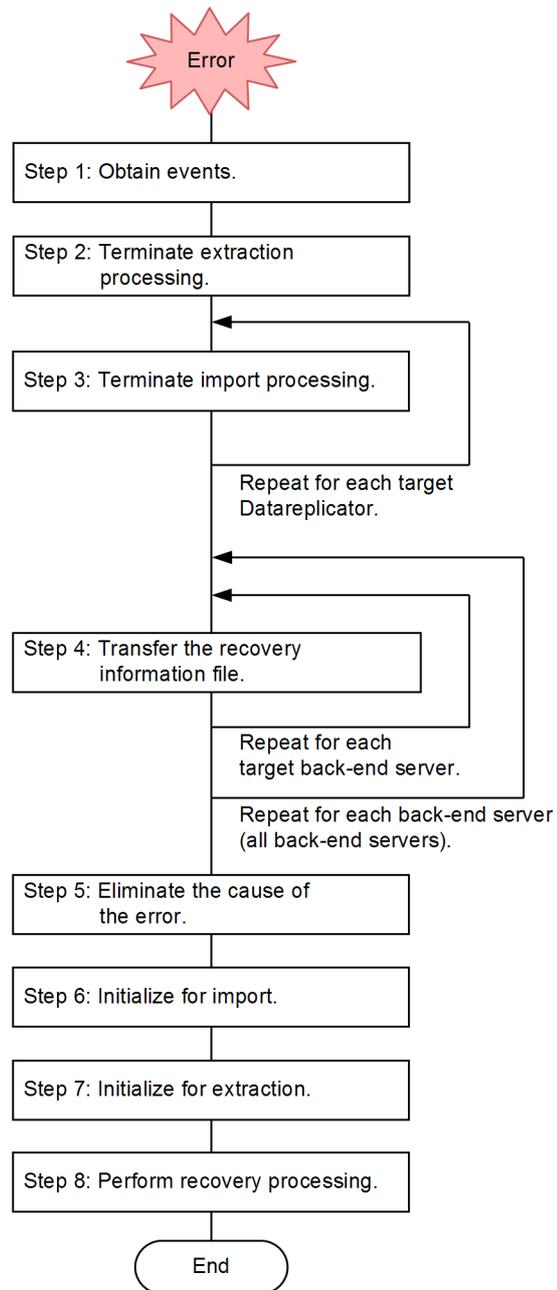
(b) Checking the capacity of the system log file

If a value other than 1 is specified in the `recover_info_send_interval` operand in the extraction system definition, allocate a sufficient capacity to the system log file so that no system log information that will be required for recovery will be overwritten by the system log during the period from occurrence of an error to the completion of recovery.

9.5.3 Overview of the recovery procedure using the system log file

The following figure provides an overview of the recovery procedure using the system log file.

Figure 9-6: Overview of the recovery procedure using the system log file



The following table provides the details of the recovery procedure using the system log

file.

Table 9-9: Details of the recovery procedure using the system log file

Item	Target	Description and example of command execution	Check item
Step 1: Obtain events	Source or target	Determine the cause of the error.	--
Step 2: Terminate extraction processing	Source	Stop the source Datareplicator. Example of command execution: <code>hdestop</code>	--
Step 3: Terminate import processing	Target	Terminate the target Datareplicator. Example of command execution: <code>hdsstop -t immediate</code>	--
Step 4: Transfer the recovery information file	Target	Transfer the recovery information file to the source in binary mode. If there are multiple recovery information files for the source system, transfer all of them to <code>\$HDEPATH</code> in the corresponding back-end servers (including back-end servers where no error occurred). If a recovery information file is a character special file for UNIX, use the <code>dd</code> command to copy it as a regular file, and then transfer the resulting recovery information file. Example of command execution: Transferring a recovery information file: <code>\$ cd \$HDSPATH</code> <code>\$ ftp source-server-name</code> <code>ftp> cd pathname-of-corresponding-HDEPATH</code> <code>ftp> binary</code> <code>ftp> put recovery-information-file-name</code> <code>ftp> bye</code> Copying a recovery information file: <ul style="list-style-type: none"> ● In AIX <code>\$ dd if=\$HDSPATH/recovery-information-file-name \</code> <code>\$ of=any-path/recovery-information-file-name \</code> <code>\$ bs=1024 skip=1</code> <code>count=recovery-information-file-size (KB)</code> ● In UNIX <code>\$ dd if=\$HDSPATH/recovery-information-file-name \</code> <code>\$ of=any-path/recovery-information-file-name \</code> <code>\$ bs=1024 count=recovery-information-file-size (KB)</code> 	--
Step 5: Eliminate the cause of the error	Source or target	Eliminate the cause of the error.	--

Item	Target	Description and example of command execution	Check item
Step 6: Initialize for import	Target	Initialize the target Datareplicator. Example of command execution: <code>hdsstart -i -f</code>	--
Step 7: Initialize for extraction	Source	Initialize the source Datareplicator. Example of command execution: <code>pdrplstop -f</code> <code>hdestart -i</code> <code>hdeprep -f extraction-definition-file-name</code> <code>pdrplstart</code>	--
Step 8: Perform recovery processing	Source	Start the source Datareplicator during data linkage recovery. Example of command execution: <code>hdestart -v</code>	Check that the KFRB05041-I message that reports completion of recovery has been output.#

Legend:

--: Not applicable

Note 1:

If an error occurs during recovery processing, Datareplicator handles the error in the same manner as when an error occurs during normal data linkage. For details about the error handling procedure, see *9.1 Error handling procedures for the source Datareplicator* and *9.2 Error handling procedures for the target Datareplicator*.

Note 2:

If the source Datareplicator is restarted during recovery processing and a recovery start message had already been output before the previous session started, the message is not output again during the restart.

If no recovery information file is specified or the contents of the recovery information file differ from the last time Datareplicator started, Datareplicator uses the contents of the recovery information file that had been specified when the recovery start message was output.

Note 3:

If either of the errors listed below occurs on the recovery information file and Datareplicator is started with the `hdestart -v` command, the KFRB05007-E message is output and Datareplicator terminates with an error. In such a case, use the unload log files to recovery the error.

9. Error Handling Procedures

1. I/O error on the recovery information file
2. The target Datareplicator has not terminated normally, but the recovery information file was allocated to the source Datareplicator during recovery processing

If error 1 occurred and the error message is output, processing resumes. However, recovery information will not be updated thereafter.

As with error 2, if the target Datareplicator has not terminated normally (for a reason such as abnormal termination or a power-loss event), the recovery information might not have been updated correctly. If an attempt is made to recover an error by using such an incomplete recovery information file, the update information that had already been imported might be retransmitted and imported again during normal data linkage processing.

#

After outputting the message, the source Datareplicator continues normal data linkage processing and sends update information.

9.6 Data linkage recovery via unload log files

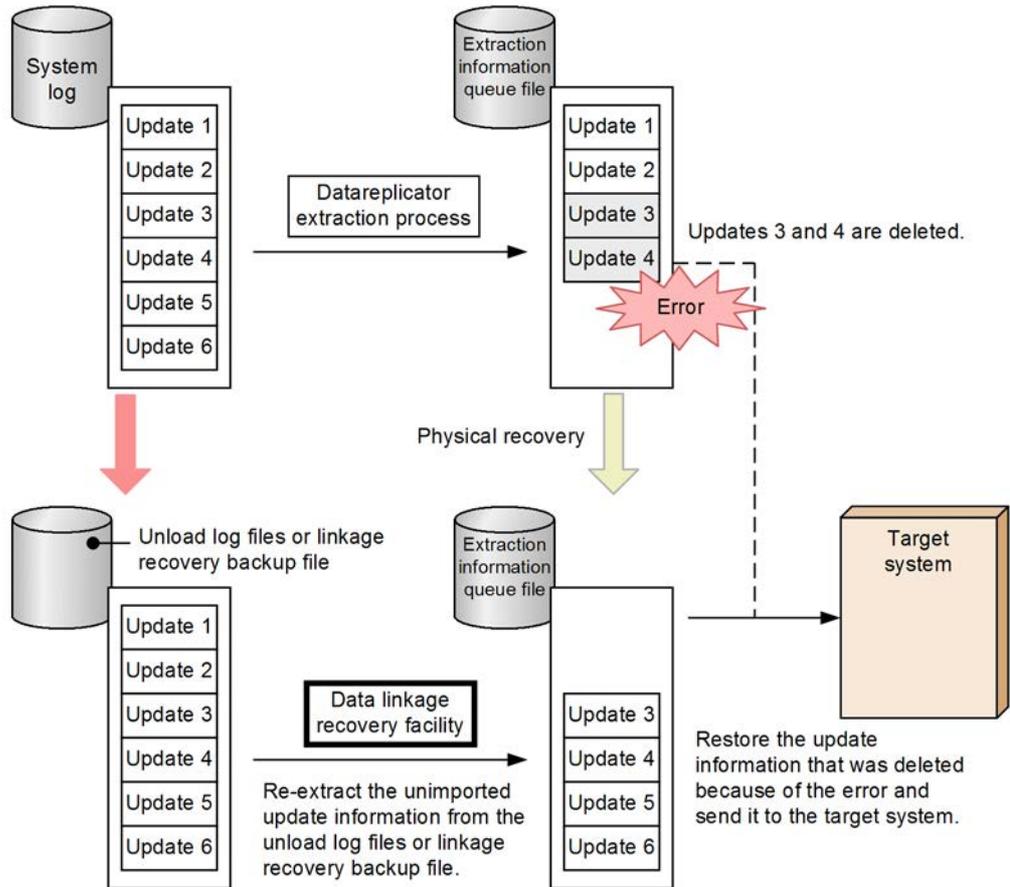
This section explains data linkage recovery via unload log files.

9.6.1 Overview of data linkage recovery via unload log files

When data linkage recovery is performed via unload log files, unimported update information that has been lost because of an error is re-extracted from the linkage recovery backup files. The linkage recovery backup files constitute a backup of the HiRDB files in the HiRDB file system area containing the unload log information; a linkage recovery backup file is created for each HiRDB file by the HiRDB file system's backup command (pdfbkup).

The following figure provides an overview of data linkage recovery via unload log files in the event of an error in the extraction information queue file.

Figure 9-7: Overview of data linkage recovery via unload log files



9.6.2 Prerequisites for data linkage recovery via unload log files

(1) Combinations of supported versions and products

The table below shows the combinations of the versions and products that support data linkage recovery via unload log files. For a product whose source system is not Datareplicator, see the applicable product documentation.

Table 9-10: Combinations of the versions and products that support data linkage recovery via unload log files

Target system		Source system	
		Datareplicator	
		06-01 or earlier	06-01-/A or later [#]
Datareplicator	04-00-/O or later	N	Y
	05-03-/D or later	N	Y
	06-01-/A or later [#]	N	Y
	07-00 or later and 08-00 or later	N	Y
	Other	N	N
XDM/DS	08-00-A	N	Y
	08-02 or later	N	Y
	Other	N	N

Y: Supports data linkage recovery via unload log files.

N: Does not support data linkage recovery via unload log files.

#

When the target of the `logmrg` command is a HiRDB large file, HiRDB Datareplicator version 06-02 or later supports data linkage recovery via unload log files.

(2) Prerequisites

Because data linkage recovery requires use of HiRDB's unload log files, all unload log files containing log information to be restored must be available.

The following table describes the causes of nonconforming data linkage and for each the applicability of data linkage recovery via unload log files (source system).

Table 9-11: Causes of nonconforming data linkage and applicability of data linkage recovery via unload log files (source system)

Product	Cause of nonconforming data linkage		Applicability	Recovery range	
Source Datareplicator	Operation error	Linkage stop instruction (definition change)	pd_rpl_init_start operand value was changed from Y to N.	Applicable	All BESs
			pd_rpl_hdepath operand was disabled.	Not applicable [#]	--
			System log file became full while the pd_log_rpl_no_standby_file_opr operand was set to continue.	Applicable	All BESs in the unit that contains the BES in which the system log file became full
		Linkage stop instruction (wrong command input interval)	pd_rpl_stop command was executed.	Applicable	All BESs
			pd_logchg -R command was executed.	Applicable	BES on which the command was executed
		Source Datareplicator was initialized		Applicable	All BESs
		Queue file became full due to execution of a large transaction.		Application not required (can be recovered by adding a queue)	--
		Source table definition was changed (DROP or CREATE TABLE). After execution, the extraction definition preprocess was not executed.		Not applicable [#]	--
		Database was updated in the no-log mode.		Not applicable [#]	--

Product	Cause of nonconforming data linkage			Applicability	Recovery range
	Error	Hardware	Disk error occurred on a file, such as a queue file.	Applicable	BES where the error occurred
		Software	Process was terminated.	Application not required (can be recovered by re-run)	BES where the process was terminated

#

You must use HiRDB Dataextractor or XDM/XT for recovery.

The following table describes the causes of nonconforming data linkage and for each the applicability of data linkage recovery via unload log files (target system).

Table 9-12: Causes of nonconforming data linkage and applicability of data linkage recovery via unload log files (target system)

Product	Cause of nonconforming data linkage			Applicability	Recovery range
Target Datareplicator	Operation error	Target Datareplicator was initialized.		Applicable	Corresponding BES
		Queue file became full due to execution of a large transaction.		Application not required (can be recovered when a queue is added by initialization)	--
	Error	Hardware	Disk error occurred on a file, such as a queue file.	Applicable	Corresponding BES
		Software	Process was terminated.	Application not required (can be recovered by re-run)	BES where the process was terminated
Target XDM/DS	Operation error	Target XDM/DS was initialized.		Applicable	Corresponding BES
		Queue file became full due to execution of a large transaction.		Application not required (can be recovered by adding a data set)	--

Product	Cause of nonconforming data linkage			Applicability	Recovery range
	Error	Hardware	Disk error occurred on a file, such as a queue file.	Applicable	Corresponding BES
		Software	Process was terminated.	Application not required (can be recovered by re-run)	--

(3) Limitations

This subsection describes limitations when data linkage recovery via unload log files is applied.

- Limitations to transactions

If an unload log file specified as input for the data linkage recovery facility is missing any part of a transaction subject to recovery, that transaction is not recovered. If the start or end of a transaction subject to recovery is missing from the input unload log files, a warning file (named `resfile_BES`) is output during recovery processing. When such a warning file is output, check its contents and add the appropriate input unload files, if necessary (for details, see 9.6.3 (2) *Preparing the unload log files*).

If a range in which data linkage was cancelled on the HiRDB system is excluded from the recovery, a transaction that spans that range is not recovered.

- Limitations to the import method at the target

If either of the following conditions is satisfied, data linkage recovery via unload log files cannot be used, in which case you must use HiRDB Dataextractor or XDM/XT for recovery:

- The mapping key is updated.

- The table-based import method is used for import processing. This is not applicable to a version that supports the import-suppress facility in the target system. For details about the import-suppress facility, see 9.6.8 *How to suppress import when the table-based import method is used for import processing at the target system*.

- Limitations to uninstallation

If you uninstall Datareplicator while the facility for performing data linkage recovery via unload log files has been set up, you will not be able to set up the data linkage recovery facility when Datareplicator is re-installed. When you uninstall Datareplicator, make sure that any setup of the facility for performing data linkage recovery via unload log files has been cancelled.

(4) Notes

- If the mapping key has been updated and a COMMIT error or a system error occurs during the target Datareplicator's last import processing, the first transaction to be recovered might result in an SQL error (SQLCODE is 100 or -803).

If this occurs, you can continue the processing by using the `skip_sqlcode` operand in the import environment definition. However, you must remember to delete this operand promptly after recovery processing has been completed.

- If `commitment_method = fxa_sqlc` is specified in the import system definition and the target Datareplicator terminates with an error or is terminated forcibly by `hdsstop -t force`, an undetermined transaction might remain in HiRDB. Use HiRDB's `pdrbk` command to determine the transaction, and then perform recovery processing.

9.6.3 Preparing for data linkage recovery via unload log files

The preparations described in this subsection are required to perform data linkage recovery via unload log files. Make sure that all environment variables whose settings are required for the target Datareplicator have been set.

(1) Creating the environment variable definition file (required for a node containing a back-end server subject to error recovery)

Under the source Datareplicator directory (`$HDEPATH`), create the file named `hde_toolenv` that defines the following environment variable. Even if you omit the environment variable, create the `hde_toolenv` file as an empty file.

```
[ TOOL_OUTPUT_DIR=execution-results-storage-directory-for-data-linkage-recovery-facility ]
```

Note: Make sure that the environment variable's specification begins in column 1.

`TOOL_OUTPUT_DIR` specifies the directory used to store the files that are output by the data linkage recovery facility (such as a transactions list). If this environment variable is omitted, `$HDEPATH` is assumed. If a relative path is specified, Datareplicator assumes the path from `$HDEPATH`.

(2) Preparing the unload log files

At the applicable back-end server, prepare the HiRDB unload log files that are to be used for the recovery of data linkage. Also from the current system log file, obtain the unload log file after swapping, if necessary.

Store the prepared HiRDB unload log files in the same directory.

(3) Checking the available disk space (required for a node that contains a back-end server subject to error recovery)

Check the following items:

- Make sure there is enough disk space for the unload log files at the target location. Also make sure there is enough disk space for the unload log files under the target directory that is specified when the `logmrg` command is executed.
- Make sure there is enough disk space for storing the transactions list (under the source Datareplicator directory) that is created when data linkage recovery facility 1 is executed. The formula for determining the size of the transactions list is as follows:

$\text{Size of transactions list (bytes)} = \text{number of transactions to be recovered} \times 64 + 4$
--

Obtain the number of transactions to be recovered from the sum of the results of executing the following command for all unload log files that are subject to recovery:

```
pdlogcat -s -kp unload-log-file-name |
  grep "Record header" | wc -l
```

- Make sure that there is about 1 megabyte of free disk space under the `$HDEPATH` directory.

(4) Deleting or moving files created during the previous execution (applicable only if recovery processing is executed more than once on the same back-end server)

When data linkage recovery via unload log files is performed, the following files are created under the directory that is specified in the `TOOL_OUTPUT_DIR` operand in the environment variable definition file (`hde_toolenv`):

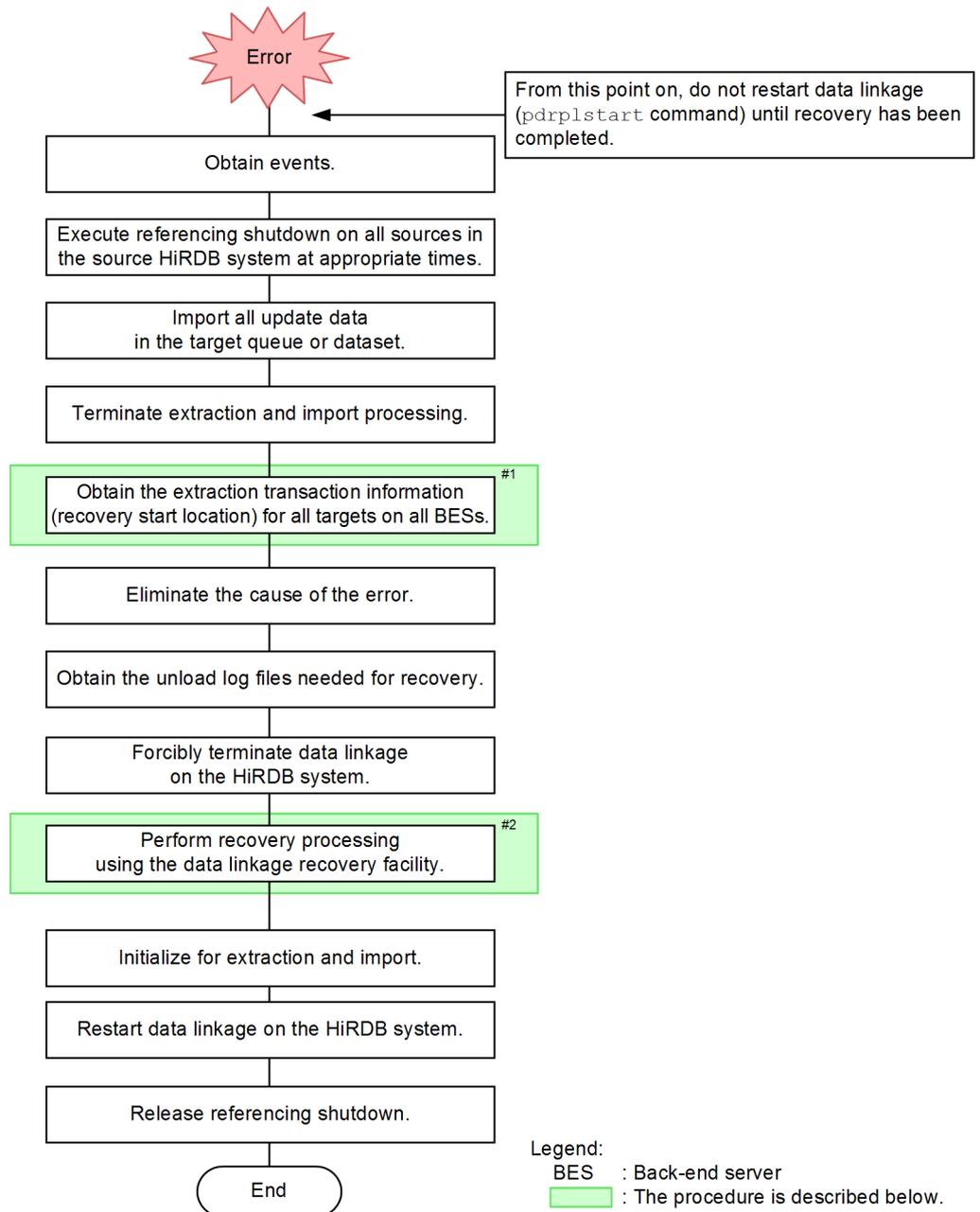
- `tranlist_BES-name`: Target transactions list file
- `res_file_BES-name`: Recovery-range unresolved transactions check file

If you execute data linkage recovery via unload log files more than once, check for the existence of these files. If they exist, either delete them or re-name them.

9.6.4 Overview of the recovery procedure using unload log files

The following figure provides an overview of the recovery procedure using unload log files.

Figure 9-8: Overview of the recovery procedure using unload log files



#1

See (1) *Obtaining the extraction transaction information (recovery start location)*.

#2

See (2) *Recovery processing using the data linkage recovery facility* and 9.6.5 *Details of data linkage recovery using unload log files*.

The following subsections describe the procedures shown in Figure 9-8 *Overview of the recovery procedure using unload log files* under *Obtain the extraction transaction information (recovery start location) for all targets on all BESs* and *Execute recovery processing using the data linkage recovery facility*.

(1) Obtaining the extraction transaction information (recovery start location)

The extraction transaction information is used to uniquely identify a transaction that has occurred on the source HiRDB system. The extraction transmission information that is output when import processing is completed indicates that all information up to that transaction executed at the source system has been imported.

To apply data linkage recovery using unload log files, you must specify the transaction in an unload log file with which recovery is to be started. By specifying the extraction transaction information existing when import processing is completed, you can recover all the transactions that were completed after that transaction (the transactions that have not been imported into the target system).

(a) How to obtain the extraction transaction information when the target is Datareplicator

The following shows how to obtain the extraction transaction information when the target is Datareplicator:

1. After importing all the update information stored in the import information queue file, terminate the target Datareplicator.
2. Search the end of the error information file for the KFRB03009-I message and obtain the extraction transaction information displayed in `Additional Transaction Info =` in the message.

If the error information file has been overwritten and there is no KFRB03009-I message, execute the command shown below to obtain the extraction transaction information from the execution results. Make sure that this command is executed before the target Datareplicator is initialized.

```
hdsrefinfmt -f target-status-file-name -l 9 -p analysis-results-output-target-file-name
```

Note: If a relative path is specified for the target status file name, Datareplicator assumes the path from \$HDSPATH.

Example of command output and how to acquire the recovery start position

When the import mode is `trn`:

```

**** HiRDB Datareplicator Reflection Infomation <Mon Nov 06 10:55:59 2006> ****
**** Status File Information ****

<Common Information[MINF]>
  Datareplicator-id:f1 Target-DB:HiRDB DB-Locale:SJIS
  Communication protocol:TCP/IP
<Replication Node Information[UJMF]>
  ReplicationNode-id:01 ReplicationType:RECIEVE EventDefinition:1,2,5,3,4
  :
  <omitted>
  :
<Reflect Entry Information[STAT]>
  HeadInformation:GroupNo=000 RunMode=Y
  Read Information:(0,6144)
  GroupInformation:GroupName=trngroup KeyRangeGroup=No
  ReflectStatusInformation:01,00,00,00,00,01
  ReflectDetailInformation:Extract System-id:01 CommitInfo:0(0),10,0
                          Locale=(EH,SJ) Source-DB=HiR Format=Sfm
                          TransactionInfo=(3,2) EventInfo=(0,0)
  Additional Transaction Info:452348bb0000000000000290
  ReflectSQLexeInformation:SQL-count<ins:1,upd:1,del:0,prg:1,cmt:2,tms:0>

```

Extraction transaction information

←

The recovery start position is Additional Transaction Info in
 <Reflect Entry Information[STAT]>
 HeadInformation:GroupNo=000.

When the import mode is `tbl`:

9. Error Handling Procedures

```

***** HiRDB Datareplicator Reflection Infomation <Fri Jan 25 12:11:22 2002> *****
***** Status File Information *****
<Common Information[MINF]>
Datareplicator-id:f4 Target-DB:HiRDB DB-Locale:SJIS
Communication protocol:TCP/IP
<Replication Node Information[UJMF]>
ReplicationNode-id:c1 ReplicationType:RECIEVE EventDefinition:1,2,5,3,4
:
  <omitted>
  :
<Reflect Entry Information[STAT]>
  HeadInformation:GroupNo=000 RunMode=N
  Read Information:(0,1024)
  GroupInformation:GroupName=trngroup KeyRangeGroup=No
  ReflectStatusInformation:10,00,00,00,00,00
  ReflectDetailInformation:Extract System-id:04 CommitInfo:0(0),1
                           Locale=(SJ,SJ) Source-DB=HiR Format=Sfm
                           TransactionInfo=(0,0) EventInfo=(0,0)
  ReflectSQLexeInformation:SQL-count<ins:0,upd:0,del:0,prg:0,cmt:0,tms:0>
<Reflect Entry Information[STAT]>
  HeadInformation:GroupNo=001 RunMode=Y
  Read Information:(0,31232)
  GroupInformation:GroupName=uoc001 KeyRangeGroup=No
  ReflectStatusInformation:01,00,00,00,00,01
  ReflectDetailInformation:Extract System-id:04 CommitInfo:3(0),1
                           Locale=(SJ,SJ) Source-DB=HiR Format=Sfm
                           TransactionInfo=(10,6) EventInfo=(1,203)
  Additional Transaction Info:3c242b8700000000000000a95
  ReflectSQLexeInformation:SQL-count<ins:58,upd:87,del:0,prg:1,cmt:6,tms:0>
<Reflect Entry Information[STAT]>
  HeadInformation:GroupNo=002 RunMode=Y
  Read Information:(0,23040)
  GroupInformation:GroupName=othergrp KeyRangeGroup=No
  ReflectStatusInformation:01,00,00,00,00,00
  ReflectDetailInformation:Extract System-id:04 CommitInfo:3(4720),1
                           Locale=(SJ,SJ) Source-DB=HiR Format=Sfm
                           TransactionInfo=(8,2) EventInfo=(0,0)
  Additional Transaction Info:3c242b870000000000000094a
  ReflectSQLexeInformation:SQL-count<ins:0,upd:0,del:0,prg:2,cmt:2,tms:0>
*****

```

Import status	How to acquire the recovery start position
All Read Information is at the same location	The recovery start position is Additional Transaction Info in <Reflect Entry Information[STAT]> HeadInformation:GroupNo=001
Read Information is at different locations	The recovery start position is Additional Transaction Info for the import group whose Read Information is delayed most [#]

#

For other import groups, `SKIP_TYPE_UNTIL` is used as the import suppression control code. For details about the specification method, see *3.3.10 (1) Using import suppression to skip errors*.

(b) How to obtain the extraction transaction information when the target is XDM/DS

The following table describes how to obtain the extraction transaction information when the target is XDM/DS:

Status of target XDM/DS	Acquisition method	Output target	Extraction transaction information
Normal or planned termination	See detailed information for the JWD396I console message that is displayed when the target XDM/DS is terminated	Console	<i>kk...k</i> part in <code>EXTTRN INF</code> : The <i>kk...k</i> that is displayed in JWD396I (24 hexadecimal characters).
	Acquire from the XDM/DS support utility's processing status output results	ABCPRINT	Information following <code>LAST-EXT-TRN-INF=</code> in the output results (24 hexadecimal characters)

(2) Recovery processing using the data linkage recovery facility

This subsection describes the *Perform recovery processing using the data linkage recovery facility* step in the recovery procedure shown in Figure 9-8 *Overview of the recovery procedure using unload log files*.

(a) Setting up for data linkage recovery

Set up the data linkage recovery facility. When the data linkage recovery facility is set up, Datareplicator's normal extraction facility is replaced with the data linkage recovery facility. Therefore, note the following points:

- Set up the data linkage recovery facility after the source Datareplicator has terminated.
- Datareplicator's extraction facility is not available while the data linkage recovery facility is set up (Datareplicator cannot extract update information on processing executed on the HiRDB system while data linkage recovery is executing). While the data linkage recovery facility is executing, do not update a table subject to data extraction.

The data linkage recovery facility can process up to the point of *extraction of update information*. After execution of the data linkage recovery facility, you must re-transmit

the update information to the target Datareplicator and execute import processing.

In the setup, execute the following two steps to extract the update information required for data linkage recovery into the extraction information queue file:

- *Data linkage recovery facility 1 (transaction retrieval phase)*

Extracts transaction information subject to recovery from the unload log segment files created by the `logmrg` command and outputs it to the target transactions list file.

- *Data linkage recovery facility 2 (extraction queue creation phase)*

Extracts update information for the transactions subject to recovery, which was extracted from the unload log segment files created by the `logmrg` command using the data linkage recovery facility 1, and stores it in the extraction information queue file.

If update information is extracted from multiple back-end servers, you must execute the recovery processing for each back-end server.

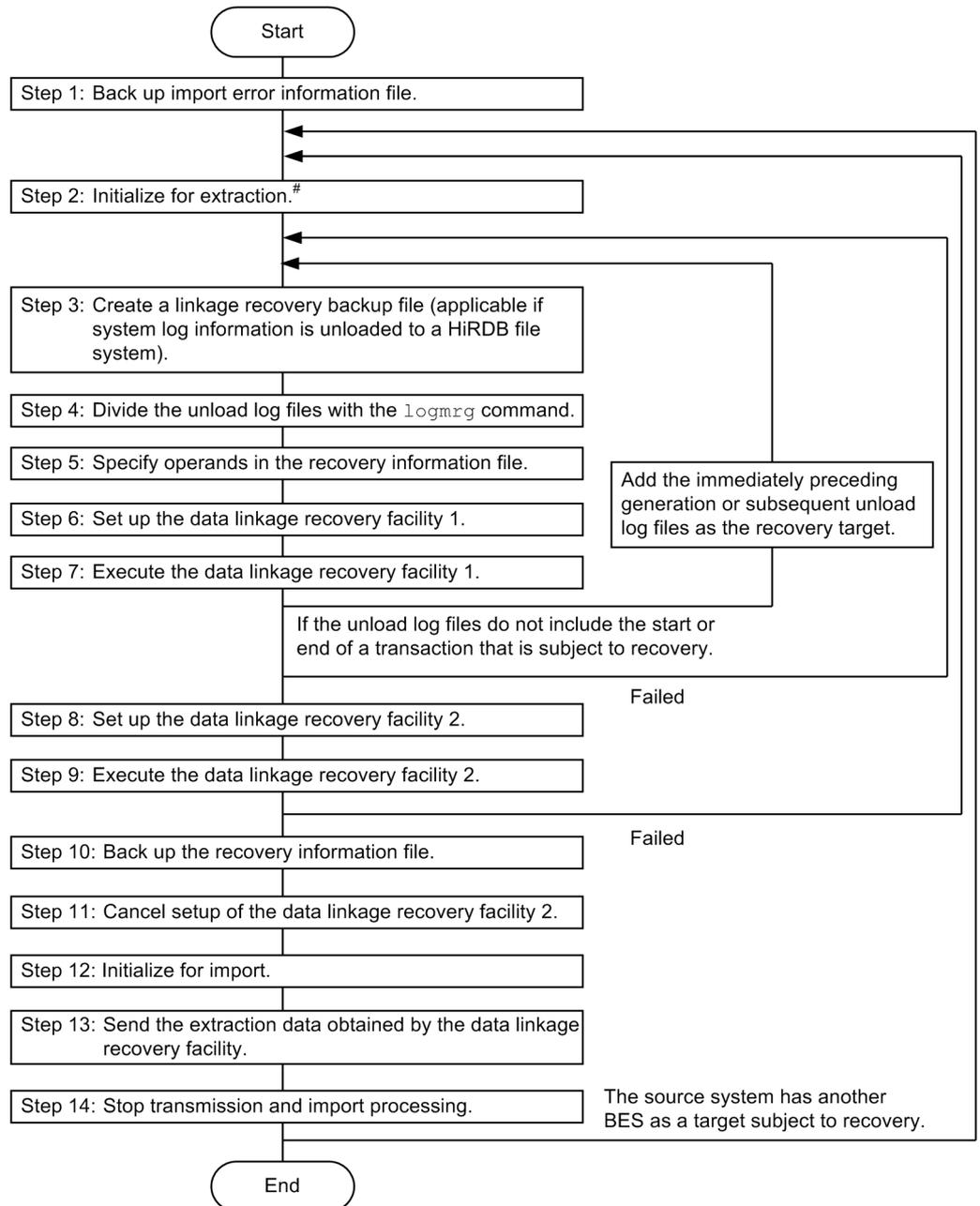
9.6.5 Details of data linkage recovery using unload log files

This subsection provides the details of the recovery procedure using unload log files.

(1) Recovery procedure using unload log files

The following figure shows the recovery procedure using unload log files.

Figure 9-9: Recovery procedure using unload log files



#

The initialization procedure for extraction includes the extraction definition preprocessing. After initialization has been completed, use the `hdeprep` command to preprocess the extraction definition. If the extraction definition is not preprocessed, a recovery error will occur.

The following provides the details of the recovery procedure using unload log files.

For the examples of command execution in the *Description and example of command execution* column, note the following:

- Command prompt `$` indicates a Datareplicator administrator's action, and `#` indicates a superuser's action (for the Windows Datareplicator, the user who installed the Datareplicator).
- The `su` command is a UNIX superuser command; it is not supported by the Windows Datareplicator.
- The `vi` command indicates the UNIX `vi` editor. In the Windows Datareplicator, use an editor such as Notepad (`notepad.exe`).

Table 9-13: Details of the recovery procedure using unload log files

Item	Target	Description and example of command execution	Input and output	Check item
Step 1: Back up the import error information file	Back-end server subject to error recovery	Save the import error information file because it contains extraction transaction information about the last information that has been imported. Example of command execution: <pre>\$ cp \$HDSPATH/errfile1 \$HDSPATH/errfile1.sav \$ cp \$HDSPATH/errfile2 \$HDSPATH/errfile2.sav</pre>	--	--
Step 2: Initialize for extraction	Manager node	Execute initialization for extraction. This includes the action preprocess. Example of command execution: <pre>\$ hdestart -i \$ hdeprep -f extfile</pre>	--	--

Item	Target	Description and example of command execution	Input and output	Check item
Step 3: Create a linkage recovery backup file (applicable when the system log information is unloaded to the HiRDB file system) ^{#1}	Back-end server subject to error recovery	Create the linkage recovery backup files in order of generations. Example of command execution: <pre>\$pdfbkup HiRDB-file-system-area -name/ HiRDB-file-name backup-file-name</pre>	Input file: HiRDB file system containing the unload log files Output file: Linkage recovery backup file	--
Step 4: Divide the unload log files with the logmrg command ^{#2}	Back-end server subject to error recovery	Execute the logmrg command. Example of command execution: <pre>\$ logmrg back-end-server-name divided-blocks-count output-target-directory input-source-directory {unload-log-file-name linkage-recovery-backup-file} [{unload-log-file-name linkage-recovery-backup-file}]</pre> <p>Note: Make sure that the unload log files or linkage recovery backup files are specified in chronological order, beginning with the oldest file.</p>	Input file: <ul style="list-style-type: none"> • Unload log file or linkage recovery backup file Output file: <ul style="list-style-type: none"> • <i>output-target-directory/output-target-directory/unload-log-file-name_sequence-number.unlog</i> • \$HDEPATH/caplogparm_BES-name 	Make sure that a file named \$HDEPATH/caplogparm_BES-name has been created and its creation date is the same as the logmrg command execution date.

9. Error Handling Procedures

Item	Target	Description and example of command execution	Input and output	Check item
Step 5: Specify operands in the recovery information file ^{#3}	Back-end server subject to error recovery	To the file named \$HDEPATH/caplogparm_ <i>BES-name</i> , add the operands indicating the recovery start location and whether recovery of HiRDB's data linkage-cancelled range is required. Example of command execution: <pre>\$ vi \$HDEPATH/caplogparm_<i>BES-name</i></pre>	Input file: <ul style="list-style-type: none"> \$HDEPATH/caplogparm_<i>BES-name</i> Output file: <ul style="list-style-type: none"> \$HDEPATH/caplogparm_<i>BES-name</i> 	--
Step 6: Set up the data linkage recovery facility 1	Node containing the back-end server subject to error recovery	Swap programs for the data linkage recovery facility 1 (transaction retrieval phase). Example of command execution: <pre>\$ su # setup_tool1 (set_tool1 in Windows Vista and Windows Server 2008)</pre>	--	Make sure that the message Setup for <hdecapture_tool1> complete is displayed.
Step 7: Execute the data linkage recovery facility 1	Manager node	Start the source Datareplicator's extraction facility. In this case, make sure that only the extraction facility is started. Example of command execution: <pre>\$ hdestart -e</pre>	Input files: <ul style="list-style-type: none"> \$HDEPATH/hde_toolenv output-target-director y/unload-log-file-name_sequence-number.unlog \$HDEPATH/caplogparm_<i>BES-name</i> Output files: <ul style="list-style-type: none"> tranlist_<i>BES-name</i> res_file_<i>BES-name</i> (applicable only when there is an incomplete transaction) 	Use the hdestate command to monitor the status until the node containing the back-end server subject to error recovery or the source Datareplicator has terminated. After termination, perform the following procedure:

Item	Target	Description and example of command execution	Input and output	Check item
				<ol style="list-style-type: none"> 1. Make sure that the error information file contains no errors. 2. Make sure that there is no file named <code>res_file_BES</code> under the directory specified in the <code>TOOL_OUTPUT_DIR</code> environment variable. 3. If only the node containing the back-end server subject to error recovery has been terminated, execute the <code>hdestop</code> command to terminate the source Datareplicator.
Step 8: Set up the data linkage recovery facility 2 ^{#4} , #5	Node containing the back-end server subject to error recovery	Swap programs for the data linkage recovery facility 2 (extraction queue creation phase). Example of command execution: <pre>\$ su # setup_tool2 (set_tool2 in Windows Vista and Windows Server 2008)</pre>	--	Make sure that the message <code>Setup for <hdcapture_tool 2> complete</code> is displayed.

9. Error Handling Procedures

Item	Target	Description and example of command execution	Input and output	Check item
<p>Step 9: Execute the data linkage recovery facility 2</p>	<p>Manager node</p>	<p>Start the source Datareplicator's extraction facility. In this case, make sure that only the extraction facility is started. Example of command execution: \$ hdestart -e</p>	<p>Input files:</p> <ul style="list-style-type: none"> • \$HDEPATH/hde_toolenv • output-target-director y/ unload-log-file-name_e_sequence-number.unlog • \$HDEPATH/caplogparm_BES-name • tranlist_BES-name <p>Output file:</p> <ul style="list-style-type: none"> • Extraction information queue file (placing the data to be recovered in queue) 	<p>Use the hdestate command to monitor the status until the node containing the back-end server subject to error recovery or the source Datareplicator has terminated. After termination, perform the following procedure:</p> <ol style="list-style-type: none"> 1. Make sure that the error information file contains no errors. 2. If only the node containing the back-end server subject to error recovery has been terminated, execute the hdestop command to terminate the source Datareplicator.
<p>Step 10: Back up the recovery information file#5</p>	<p>Back-end server subject to error recovery</p>	<p>Rename the file \$HDEPATH/caplogparm_BES-name. Example of command execution: \$ mv \$HDEPATH/caplogparm_BES-name \$HDEPATH/caplogparm_BES-name.sav</p>	<p>--</p>	<p>This step protects from a malfunction if the user forgets to uninstall the facility.</p>
<p>Step 11: Cancel the setup of data linkage recovery facility 2</p>	<p>Node containing the back-end server subject to error recovery</p>	<p>Execute the unsetup_tool command. Example of command execution: \$ su # unsetup_tool (unset_tool in Windows Vista and Windows Server 2008)</p>	<p>--</p>	<p>Make sure that the message Unsetup for <hdecapture_tool> complete is displayed.</p>

Item	Target	Description and example of command execution	Input and output	Check item
Step 12: Initialize for import	Manager node	Initialize for import processing. Example of command execution: \$ hdsstart -i	--	--
Step 13: Send the extraction data obtained by the data linkage recovery facility	Manager node	Start the source Datareplicator's transmission facility. In this case, make sure that only the transmission facility is started. Example of command execution: \$ hdestart -s \$ hdestate	--	After starting the transmission facility, use the <code>hdestate</code> command to monitor when the transmission queue file's <code>read</code> offset reaches the <code>write</code> offset. When the <code>read</code> offset has reached the <code>write</code> offset, execute the <code>hdestop</code> command to terminate the source Datareplicator.
Step 14: Stop transmission and import processing	Manager node	Terminate the source Datareplicator's transmission process and the import process. Example of command execution: \$ hdestop \$ hdsstop	--	--

#1

To check the date of HiRDB files, execute the following command:

```
pdlogcat_s -i backup-source-HiRDB-file | grep "First use"
```

The *backup-source-HiRDB-file* is the HiRDB file from which the linkage recovery backup file was created. Specify the file in the format *HiRDB-file-system-area/HiRDB-file-name*.

#2

To check the date of an unload log file, execute the following command:

```
pdlogcat_s -i unload-log-file-name
```

In the command execution results, the date displayed in the row `First use` is the date the update log was actually stored in the corresponding unload log file. Make

sure that the unload log files specified in the `logmrg` command are in chronological order of this date.

#3

Add the parameters at the end of the file according to the following rules:

Operand name		Classifi- cation	Setting	Description
RCVR_START	Earlier than 06-02	Optional	<p>You can specify a maximum of 4,096 RCVR_START operands.</p> <p>Format</p> <p><i>extraction-transaction-information</i></p> <ul style="list-style-type: none"> <i>extraction-transaction-information</i> ~ <24 hexadecimal characters> <p>Specifies the extraction transaction information that indicates the start location for error recovery. For details about how to obtain the extraction transaction information, see <i>9.6.4(1) Obtaining the extraction transaction information (recovery start location)</i>.</p>	If this operand is omitted, Datareplicator assumes that the beginning of the first unload log file that is read is the start location for error recovery.
	06-02 or later	Optional	<p>You can specify a maximum of 4,096 RCVR_START operands.</p> <p>Format</p> <p>{<i>target-identifier-number</i> * } [, <i>extraction-transaction-information</i>]</p> <ul style="list-style-type: none"> <i>target-identifier-number</i> <p>Specifies the target identifier number specified in the extraction system definition. If an asterisk (*) is specified as the target identifier number, all transmission targets are assumed. If only the target identifier number is specified in this operand, Datareplicator assumes the beginning of the input unload log file as the recovery start location.</p> <p>If an asterisk (*) is specified as the target identifier number in this operand and this operand is specified again, an error results. Specifying the same target identifier number more than once also results in an error.</p>	<p>For each target, specify the <i>transmission target identifier subject to error recovery</i> and the <i>extraction transaction information indicating the location of error recovery</i>.</p> <p>If this operand is omitted, Datareplicator assumes that all transmission target identifiers are to be recovered and that the beginning of the input unload log file is the recovery start location.</p>

Operand name	Classification	Setting	Description
		<ul style="list-style-type: none"> <i>extraction-transaction-information</i> ~ <24 hexadecimal characters> For details about how to obtain the extraction transaction information, see 9.6.4(1) <i>Obtaining the extraction transaction information (recovery start location)</i>. 	
RCVR_RPLSTOP	Optional	Y: HiRDB's linkage-cancelled range is subject to recovery. N or a value other than Y: HiRDB's linkage-cancelled range is not subject to recovery.	If the error recovery range includes HiRDB's linkage-cancelled range, specify whether this linkage-cancelled range is to be subject to recovery. When this operand is omitted, N is assumed.

Notes about specifying the RCVR_START operand

You can recover the input unload log files only when the mapping key has not been updated (if the mapping key is updated, nonconformity of data linkage might occur due to duplicated updating during recovery). Additionally, to avoid duplicated update errors at the target, you must set `SQLCODE` for the missing key error (100) and duplicate key error (-803) in the `skip_sqlcode` operand in the import environment definition prior to the recovery.

Example (added parameters are underlined)

```

INPUT = /HiRDBDS/hirdbb/HDE/work/unldlog2_1.unlog,10,UNLDLOG
INPUT = /HiRDBDS/hirdbb/HDE/work/unldlog2_2.unlog,10,UNLDLOG
INPUT = /HiRDBDS/hirdbb/HDE/work/unldlog2_3.unlog,10,UNLDLOG
BLOCKBUF = 22200
RCVR_START = SND01,3BA6E580000000000000000015
RCVR_RPLSTOP = Y

```

#4

If there is a file named `res_file_BES-name`, check the file contents and take appropriate action according to the following (this is a text file):

Output information	Description	Action
N.G (start nothing)	The unload log information contains data that is missing the start of a transaction.	Add the immediately preceding generation of unload log file, and then re-execute the procedure starting with step 3. During re-execution, either delete the file named <code>res_file_BES-name</code> or save the file under a different name.
N.G (end nothing)	The unload log information contains data that is missing the end of a transaction.	Make sure that referencing shutdown is in effect on all RDAREAs that contain the source table. When referencing shutdown is in effect: Ignore this output and resume the subsequent recovery procedure. When referencing shutdown is not in effect: Apply referencing shutdown, add the unload log files up to the current point, then re-execute the procedure starting with step 3. During re-execution, either delete the file named <code>res_file_BES-name</code> or save the file under a different name.
N.G (rplstop found) N.G (start nothing, rplstop) N.G (end nothing, rplstop)	There is a transaction within HiRDB's data linkage-cancelled range.	This output information means that HiRDB's data linkage-cancelled range is not subject to recovery. If there is no problem excluding HiRDB's data linkage-cancelled range from recovery processing, ignore this message and resume the subsequent recovery procedure. To include HiRDB's data linkage-cancelled range in the recovery processing, specify <code>RCVR_RPLSTOP=Y</code> , and then re-execute the procedure starting with step 6.

#5

If the node containing the back-end server subject to error recovery also contains other back-end servers, the recovery tool is also executed on those back-end servers. In this case, an error such as missing files (required for recovery) might be detected on back-end servers that are not subject to error recovery (see footnote 3 above). Ignore such errors.

(2) Handling errors during data linkage recovery via unload log files

If any errors occur during data linkage recovery via unload log files, Datareplicator outputs the `KFRB05009-E` message to the error information file for all errors related to data linkage recovery via unload log files. Identify the nature of any error based on the information that is displayed in the `function:` section of the message. The following table explains the `function:` section.

Table 9-14: Contents of the function: section in the KFRB05009-E message

Information in the function: section	Description	Action
analyze error	An analysis error occurred in the \$HDEPATH/caplogparm_BES-name file.	Check to see if there is a file named \$HDEPATH/caplogparm_BES-name. If there is no such file, re-execute the procedure starting with step 3. If the file exists, the operand added to the corresponding file is invalid. Correct the operand, and then re-execute the procedure starting with step 7.#
env open_error	An open error occurred in the environment variable definition file (hde_toolenv).	Check to see if the environment variable definition file has been created under \$HDEPATH. If the file has not been created, create it, and then re-execute the procedure starting with step 7.#
inttrn error	Initialization of the recovery tool resulted in an error.	This error occurs only when there is not enough memory. Terminate another program to increase the available memory, and then re-execute the procedure starting with step 7.#
open_error	An open error occurred in an unload log segment file.	The unload log segment files might not have been created correctly by the logmrg command. Re-execute the procedure starting with step 3.#
read_error	A read error occurred in an unload log segment file.	The unload log segment files might not have been created correctly by the logmrg command. Re-execute the procedure starting with step 3.#
trnout error	An output error occurred in the transactions list.	A space shortage might have occurred in the output target directory for the transactions list. Change the output target directory, and then re-execute the procedure starting with step 7.#
trnget error	An input error occurred in the transactions list.	The transactions list has not been created correctly. Execute step 11#, and then re-execute the procedure starting with step 6.#
blk_invalid	An invalid log block was detected in the unload log file.	This is an internal conflict error. Contact the developer.

Information in the function: section	Description	Action
start point err	Specified extraction transaction information is invalid.	Invalid extraction transaction information was specified in the RCVR_START operand, or some required unload log files were not specified for input. If the specified extraction transaction information is invalid, correct it, and then re-execute the procedure starting with step 7;# if required unload log files are missing, add the files, and then re-execute the procedure starting with step 3.#
seq_invalid	The order of log blocks is invalid in the unload log files	The order of the unload log files or linkage recovery backup files specified during execution of the logmrg command might be invalid (not sorted in ascending order of the log output date). Check the order of the input unload log files or linkage recovery backup files, and then re-execute the procedure starting with step 3.#
param len error	Length of the directory path name specified in the TOOL_OUTPUT_DIR environment variable is invalid.	Correct the value, and then re-execute the procedure starting with step 7.#
remain trnentry	The number of transactions in the transactions list created by the data linkage recovery facility 1 does not match the number of transactions recovered by the data linkage recovery facility 2.	This is an internal conflict error. Contact the developer.

#

See (1) Recovery procedure using unload log files.

9.6.6 Commands provided by the data linkage recovery facility

You use the following commands for data linkage recovery via unload log files:

Command name	Description
logmrg	Divides unload log files subject to recovery, or linkage recovery backup files into smaller files in such a format that the error recovery tools can read them.
setup_tool1#	Sets up the data linkage recovery facility 1.
setup_tool2#	Sets up the data linkage recovery facility 2.

Command name	Description
unsetup_tool#	Disables the data linkage recovery facility that has been set up and enables Datareplicator's extraction facility.

#

The command names differ in Windows Vista and Windows Server 2008, as shown in the following table:

OS other than Windows Vista or Windows Server 2008	Windows Vista and Windows Server 2008
setup_tool1	set_tool1
setup_tool2	set_tool2
unsetup_tool	unset_tool

(1) logmrg command

Execute the logmrg command in a directory for which the logon user has access privilege. An access privilege error occurs if the command is executed in a directory for which the logon user does not have access privilege.

Format

```
logmrg HiRDB-server-name
       divided-blocks-count
       output-target-directory
       input-source-directory
       { HiRDB-unload-log-file-name | linkage-recovery-backup-file-name }
       [ { HiRDB-unload-log-file-name | linkage-recovery-backup-file-name } ]
```

Options

HiRDB-server-name

Specify the HiRDB server name.

divided-blocks-count

Specify the number of blocks for dividing a HiRDB unload log file. The relationship between this value and the maximum size of an unload log segment file is as follows:

<p><i>Maximum size of unload log segment file</i> (bytes) = <i>pd_log_max_data_size operand value for HiRDB subject to recovery</i> x <i>number of blocks</i></p>
--

We recommend that you specify in this operand a value in the range 100 to 1000.

Make sure that the size of an unload log segment file does not exceed 2 GB - 1 byte.

output-target-directory

Specify the absolute path of the storage directory for the `logmrg` command's output files (such as the unload log segment files). The permitted maximum length of the path name is 128 bytes. Do not specify the directory path of a name that contains a space.

input-source-directory

Specify the name of the storage directory for the input HiRDB unload log files. The permitted maximum length of the name is 128 bytes. Do not specify the directory path of a name that contains a space.

If you use Datareplicator version 07-02 or later, you can specify a large file as the input load log file or linkage recovery backup file. Note that an unload log segment file to be output cannot be a large file.

HiRDB-unload-log-file-name | linkage-recovery-backup-file-name

~ <symbolic name of 1-32 characters>

Specify an input HiRDB unload log file name or linkage recovery backup file name. Specify file names in the order of generations. Note that an unload log segment file must be a regular file or a non-large file.

To check the date of an unload log file, execute the following command:

```
pdlogcat_s -i unload-log-file-name | grep "First use"
```

To check the date of a linkage recovery backup file, execute the command:

```
pdlogcat_s -i backup-source-HiRDB-file-name | grep "First use"
```

The backup source HiRDB file is the HiRDB file from which the linkage recovery backup file was created. Specify it in the format *HiRDB-file-system-area-name/HiRDB-file-name*.

Messages

Message	Action
[aa...aa] File (<i>file-name</i>) open error, errno=xx...xx	Identify the cause of the error on the basis of the displayed <code>errno</code> , eliminate it, then re-execute the command.
[aa...aa] File size after dividing exceeds 2 GB, filesize = bb...b	Size of the unload log segment file exceeds 2 GB. Specify a smaller number of partitioned blocks in the <code>logmrg</code> command argument, and then re-execute the command.
[aa...aa] Interface error occurred	Contact the customer support center.

Message	Action
[aa...aa] Invalid file format	Re-execute the command specifying unload log files or linkage recovery backup files as the input files.
[aa...aa] Lseek error, errno= xx...xx	Identify the cause of the error on the basis of the displayed <code>errno</code> , eliminate it, then re-execute the command.
[aa...aa] Malloc error, size= xx...xx	Add more memory, and then re-execute the command.
[aa...aa] Number of HiRDB files in backup file exceeds <i>x</i>	A specified linkage recovery backup file contains information about more than <i>x</i> HiRDB files. Specify a linkage recovery backup file that was created for a HiRDB file.
[aa...aa] Read error, errno= xx...xx	Identify the cause of the error on the basis of the displayed <code>errno</code> , eliminate it, then re-execute the command.
[aa...aa] Stat error, errno= xx...xx	Identify the cause of the error on the basis of the displayed <code>errno</code> , eliminate it, then re-execute the command.
[aa...aa] system call error. errno=xx...xx	Identify the cause of the error on the basis of the system call name and the displayed <code>errno</code> , eliminate it, then re-execute the command.
[aa...aa] Write error, errno= xx...xx	Identify the cause of the error on the basis of the displayed <code>errno</code> , eliminate it, then re-execute the command.
File open error. errno=xx...xx file=yy...yy	Identify the cause of the error on the basis of the displayed <code>errno</code> , eliminate it, then re-execute the command.
HDEPATH env not set.	Set the HDEPATH environment variable, and then re-execute the command.
HDEPATH env length invalid	Correct the settings of the HDEPATH environment variable, and then re-execute the command.
Invalid command argument. Usage: logmrg <server name> <block numbers> <output directory> <input directory> <unload log file name> . . .	Correct the operand specification, and then re-execute the command.
Invalid logcut output file.	Identify the cause of the error on the basis of the message that was issued before this message, eliminate it, then re-execute the command.
Specified output directory is not full path name.	Correct the operand specification, and then re-execute the command.

(2) *setup_tool1* command

The `setup_tool1` command sets up an environment for the data linkage recovery facility 1. At the time of setup, the command renames the program to be used for

normal operation, and then copies the data linkage recovery facility 1.

Format

```
setup_tool1 (set_tool1 in Windows Vista and Windows Server 2008)
```

Messages

Message	Action
Setup for <hdecapture_tool1> complete.	Setup of the data linkage recovery facility 1 has been completed. Continue with the recovery procedure.
Setup for <hdecapture_tool1> failed.	Datareplicator's load module might be invalid. Re-install Datareplicator, and then re-execute the command.
<hdecapture> backup failed.	Datareplicator's load module might be invalid. Re-install Datareplicator, and then re-execute the command.
Setup for <hdecapture_tool1> already execute.	The data linkage recovery facility 1 has already been set up.
HiRDB Datareplicator is not installed.	Datareplicator's load module might be invalid. Re-install Datareplicator, and then re-execute the command.
Please execute by root user.	Re-execute the command by the administrator privilege (superuser).

(3) *setup_tool2 command*

The `setup_tool2` command sets up an environment for the data linkage recovery facility 2. At the time of setup, the command renames the program to be used for normal operation, and then copies the data linkage recovery facility 2.

Format

```
setup_tool2 (set_tool2 in Windows Vista and Windows Server 2008)
```

Messages

Message	Action
Setup for <hdecapture_tool2> complete.	Setup of the data linkage recovery facility 2 has been completed. Continue with the recovery procedure.
Setup for <hdecapture_tool2> failed.	Datareplicator's load module might be invalid. Re-install Datareplicator, and then re-execute the command.

Message	Action
<hdecapture> backup failed.	Datareplicator's load module might be invalid. Re-install Datareplicator, and then re-execute the command.
Not setup for <hdecapture_tool1>.	Execute the command after setting up the data linkage recovery facility 1.
HiRDB Datareplicator is not installed.	Datareplicator's load module might be invalid. Re-install Datareplicator, and then re-execute the command.
Please execute by root user.	Re-execute the command by the administrator privilege (superuser).

(4) *unsetup_tool* command

The *unsetup_tool* command cancels the environment setup for the data linkage recovery facility 1 or 2. At the time of *unsetup*, the command renames the facility that has been set up and copies the programs that are used for normal operation.

Format

```
unsetup_tool (unset_tool in Windows Vista and Windows Server 2008)
```

Messages

Message	Action
Unsetup for <hdecapture_tool> complete.	Setup of the data linkage recovery facility has been cancelled.
Unsetup for <hdecapture_tool> failed. Please re-install HiRDB Datareplicator.	Datareplicator's load module might be invalid. Re-install Datareplicator, and then re-execute the command.
Not setup for <hdecapture_tool>.	The data linkage recovery facility has not been set up.
Please execute by root user.	Re-execute the command by the administrator privilege (superuser).

9.6.7 Procedure after execution of data linkage recovery facility

(1) *Procedure after completion of recovery*

Delete the following files once data linkage recovery via unload log files has been completed:

- Unload log segment files

Delete the *output-target-directory/*

input-unload-log-file-name_sequence-number.unlog that are created by the logmrg command and the files \$HDEPATH/caplogparm_*BES-name* (or their aliases if aliases is used).

- Data linkage recovery facility-related files

Delete the following files in the directory that was specified in the TOOL_OUTPUT_DIR environment variable in the environment variable definition file (hde_toolenv) under the source Datareplicator directory (\$HDEPATH):

- tranlist_*BES-name*, res_file_*BES-name*, and hde_toolenv

(2) Checking data linkage recovery

Check the following items to determine whether data linkage recovery executed correctly:

- Number of data items in the source and target tables

Obtain the number of data items in the source table and the target table and verify that the values match.

- Data in the source and target tables

Sample some data in the recovery range from the source and target tables to verify that the data matches.

9.6.8 How to suppress import when the table-based import method is used for import processing at the target system

If the target system employs the table-based import method, you can recover data linkage by using import suppression to skip errors.

You can use import suppress to skip errors in the following cases:

- No transaction-based or table-based import event is used during recovery.
- The extraction definition has not been changed (including re-execution of the hdeprep command).

For the import-suppress control code for data linkage recovery, use SKIP_TYPE_UNTIL. For details about using import suppression to skip errors, see *3.3.10 (1) Using import suppression to skip errors*.

(1) How to create an import suppression list file

This subsection describes how to create an import suppression list file.

Make sure that all update information stored in the import information queue files has been imported.

1. Terminate the source Datareplicator.
2. Execute the hdsrefinfm command and obtain the extraction transaction

information from the execution results for each group.

Execute the `hdsrefinfmt` command as follows without specifying an import group name:

```
hdsrefinfmt -f import-status-file-name -l 9 -p analysis-results-output-file-name
```

3. Create an import suppression list file under `$HDSPATH` on the basis of the extraction transaction information.

The following example shows the correspondence between the output of the `hdsrefinfmt` command and the import-suppress control list file, where the import mode is `trn`:

(2) Notes

- If you terminate an import process after using `SKIP_TYPE_UNTIL` to suppress import, make sure that this specification is deleted. If the import process is restarted without deleting this specification, import of all detected update information will be suppressed.
- When you specify `SKIP_TYPE_UNTIL`, do not switch the import mode or change the source Datareplicator's definitions. Because the import process terminates and starts automatically, the processed `SKIP_TYPE_UNTIL` takes effect again and import suppress might be executed illegally.
- Update information suppressed by `SKIP_TYPE_UNTIL` cannot be recovered with the error recovery facility. You must use `HiRDB Dataextractor` and `XDM/XT` to recover the information.

9.7 Facility for recovering the extraction information queue file

This section explains the facility for recovering the extraction information queue file.

9.7.1 Overview of the facility for recovering the extraction information queue file

If an error occurs only in the extraction information queue file for a reason such as a disk failure, the facility for recovering the extraction information queue file re-creates the extraction information queue file based on the information in the extraction server status file.

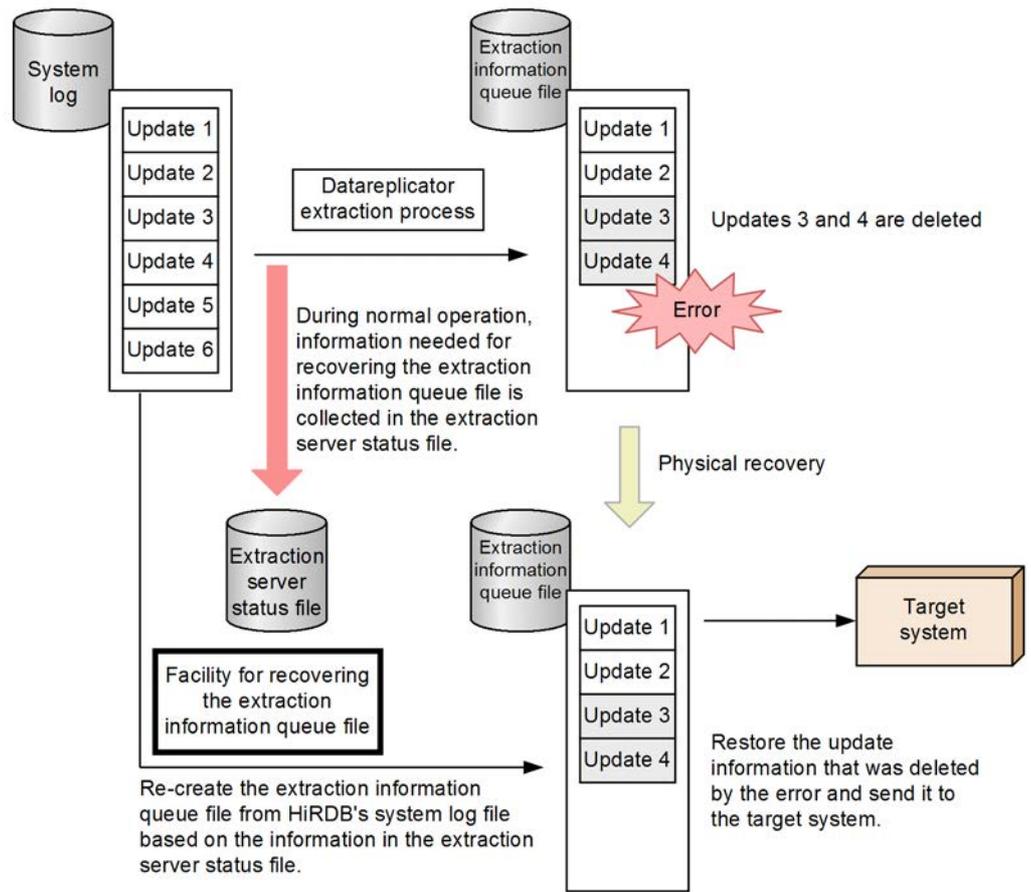
The facility for recovering the extraction information queue file provides the following benefits:

- There is no need to terminate applications (updating applications) at the source.
- The recovery procedure is simple.

Note that this facility cannot be used if the system log file has been overwritten.

The following figure provides an overview of the facility for recovering the extraction information queue file.

Figure 9-10: Overview of the facility for recovering the extraction information queue file



9.7.2 Prerequisites for the facility for recovering the extraction information queue file

This subsection explains the prerequisites for using the facility for recovering the extraction information queue file.

(1) Prerequisites

The following conditions must be satisfied:

- The source database is HiRDB.

- No error has occurred in any of the following files and these files have not been initialized:
 - Extraction master status file
 - Extraction server status file
 - Data linkage file
- HiRDB's system log file has not been overwritten.
- The extraction information queue file is not full.
- The `hdeprep` command was not executed during the extraction processing performed before the error occurred.
- HiRDB's `pdrplstop` command was not executed during operations prior to recovery.
- The `syncterm` operand value in the extraction system definition is not `true`.

(2) Initial start of the source Datareplicator (during upgrading)

If you have upgraded the source Datareplicator from a version earlier than 08-01 to 08-01 or later, perform an initial start on the source Datareplicator.[#]

An extraction information queue file whose version is earlier than 08-01 cannot be recovered by the facility for recovering the extraction information queue file (instead, use the data linkage recovery facility). If there is an extraction information queue file whose version is earlier than 08-01, the `KFRB05037-W` message will be output when the source Datareplicator starts.

#

If your system does not use the facility for recovering the extraction information queue file, there is no need to perform an initial start.

(3) Notes about using Datareplicator file system areas

If the extraction information queue file is stored in a Datareplicator file system area and the extraction information queue file is initialized, other files stored together with the extraction information queue file (extraction server status file or data linkage file) will also be initialized. If an error occurs while the facility for recovering the extraction information queue file is running, it is only necessary to restore the extraction server status file from its backup.

For this reason, it is advisable to store the extraction information queue file, the extraction server status file, and the data linkage file in separate Datareplicator file system areas.

For a HiRDB/Parallel Server, store the extraction information queue file for each of multiple back-end servers in a separate Datareplicator file system area.

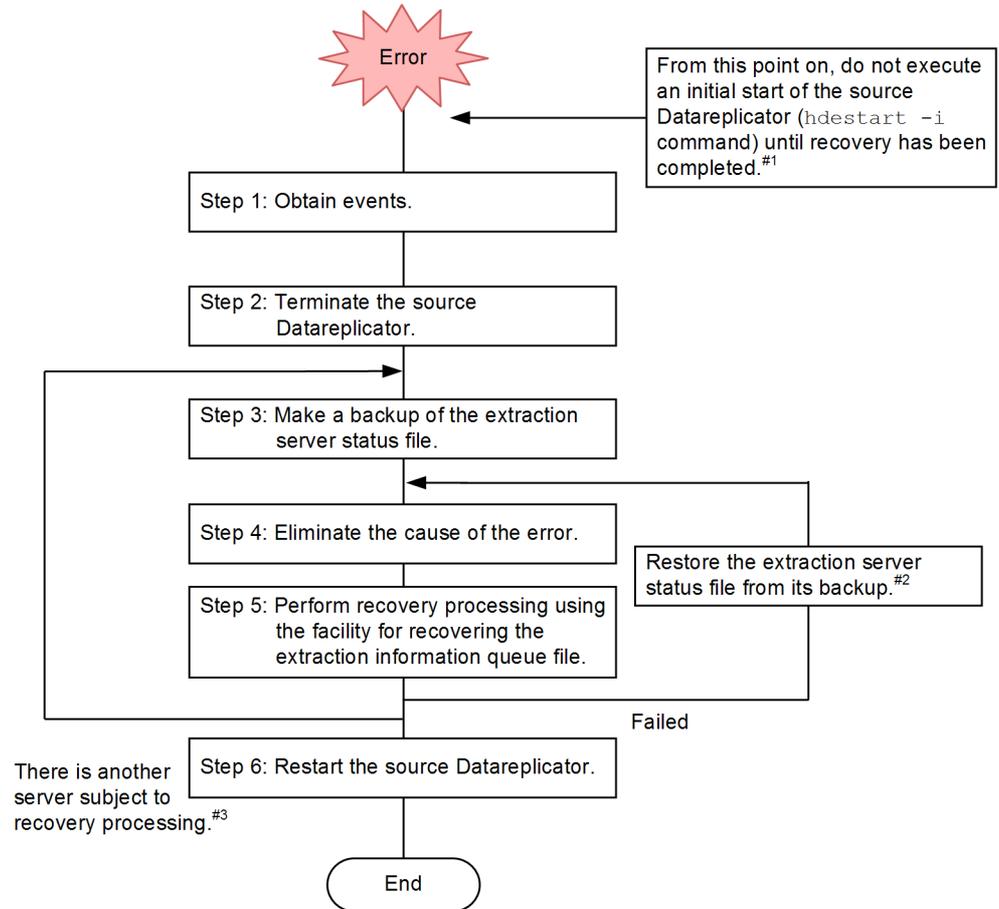
(4) Limitations

If an error occurs in the extraction information queue file while the data linkage recovery facility is running, the facility for recovering the extraction information queue file cannot be used. In such a case, use the data linkage recovery facility to recover the error.

9.7.3 Recovery procedure using the facility for recovering the extraction information queue file

The following figure shows the recovery procedure using the facility for recovering the extraction information queue file.

Figure 9-11: Recovery procedure using the facility for recovering the extraction information queue file



#1

If an initial start is executed, the facility for recovering the extraction information queue file can no longer be used because other files, including status files, are initialized. If you have executed an initial start, synchronize the source and target databases, and then execute an initial start on the source and target Datareplicators.

#2

Restore only the extraction server status file from its backup. Do not restore the data linkage file because this file is being used by HiRDB.

#3

If the source system is a HiRDB/Parallel Server, perform recovery processing for each back-end server. If multiple back-end servers are subject to recovery processing, repeat steps 3 through 5 for each back-end server that is subject to recovery processing.

The following table provides the details of the recovery procedure.

Table 9-15: Details of the recovery procedure using the facility for recovering the extraction information queue file

Item	Target	Description and example of command execution	Check item
Step 1: Obtain events. #1	Server subject to error recovery	Determine the cause of the error. If the error occurred only in the extraction information queue file, use the procedure described here to recover the error.	--
Step 2: Terminate the source Datareplicator.	Manager node	Stop the source Datareplicator. Example of command execution: hdestop	--
Step 3: Make a backup of the extraction server status file.	Server subject to error recovery	Make a backup of the extraction server status file. Example of command execution: See 6.4.2(7)(b) <i>Backing up the extraction server status file.</i>	--

Item	Target	Description and example of command execution	Check item
Branch step inclusion if Step 5 fails: If the initial recovery failed and a second recovery is attempted.	Server subject to error recovery	Restore the extraction server status file from its backup. Example of command execution: <i>See 6.4.2(7)(b) Backing up the extraction server status file.</i>	--
Step 4: Eliminate the cause of the error.	Server subject to error recovery	Eliminate the cause of the error in the extraction information queue file.	--
Step 5: Perform recovery processing using the facility for recovering the extraction information queue file.	Manager node	Recover the extraction information queue file. Example of command execution: <code>hdestart -R -k queue -b bes1</code>	Make sure that the range of files subject to recovery processing that is displayed in the KFRB05034-I message when recovery starts is the same in the KFRB05035-I message when recovery is completed. Use the <code>hdestate</code> command to monitor the source Datareplicator until it terminates. ^{#2} Alternatively, verify that the KFRB00510-I message indicating termination of the source Datareplicator has been output to the manager node's extraction master error information file. After the source Datareplicator has terminated, verify that no errors have been output to the error information file.
Step 6: Restart the source Datareplicator.	Manager node	Restart the source Datareplicator. Example of command execution: <code>hdestart</code>	--

Legend:

--: Not applicable

#1

If HiRDB's system log file has wrapped around and information needed for recovery has been overwritten, the facility for recovering the extraction information queue file cannot be used (in which case the KFRB05011-E message is output). In such a case, use the data linkage recovery facility.

#2

You cannot execute the `hdestart` or the `hdestop` command until the source Datareplicator has terminated.

(1) Recovery processing using the facility for recovering the extraction information queue file

The facility for recovering the extraction information queue file initializes the extraction information queue file, and then restores it.

(a) Initializing the extraction information queue file

To restore an extraction information queue file in which an error has occurred, you must first initialize the affected extraction information queue file.

The facility for recovering the extraction information queue file begins by initializing all extraction information queue files.

(b) Recovering the extraction information queue file

Restore the extraction information queue file if all extraction processing had been completed before the error occurred and there is update information that has not been sent to the target Datareplicator.

Restoration of the extraction information queue file involves extracting the update information that had been extracted before the error occurred from the system log file based on the extraction server status file, and then storing that update information in the extraction information queue file.

(2) Restarting the source Datareplicator

After you have restored the extraction information queue file, restart the source Datareplicator. When the source Datareplicator restarts, the update information re-extracted by the facility for recovering the extraction information queue file is sent to the target Datareplicator. Extraction processing is resumed for update information that was generated after the error occurred.

For an extraction information queue file whose update information had all been sent to the target Datareplicator before the error occurred, extraction processing is restarted from the initialized status.

(3) If an error occurs during restoration of the extraction information queue file

If an error occurs during restoration of the extraction information queue file, restore the extraction server status file from its backup, and then eliminate the cause of the error by referencing *9.1.2 Error handling methods*. After that, re-execute the facility for recovering the extraction information queue file.

Note that if an error occurs in any of the files listed below, the facility for recovering the extraction information queue file can no longer be used. In such a case, use the data linkage recovery facility.

- Extraction master status file
- Extraction server status file
- Data linkage file

9.8 Acquisition of untransmitted information due to import errors (update-SQL output facility)

With Datareplicator, there is a time delay between updating of the source database and updating of the target database. If an error occurs at the source database during import processing, inconsistency might occur between the source and target databases.

The update-SQL output facility outputs the information from the source database that has not been transmitted (information that has not been imported into the target database) as SQL statements. This facility enables you to determine the unimported information.

Note:

While the update-SQL output facility is being used, update information cannot be extracted or sent to the target system.

Using the update-SQL output facility will not clear the untransmitted information from the extraction information queue files.

(1) How to use

This subsection describes how to use the update-SQL output facility.

(a) Prerequisites

To use the update-SQL output facility, the following conditions must be satisfied:

- The source Datareplicator's processes can be started (recovered from errors such as CPU errors).
- The following files in the source system are in normal status (recovered from errors such as disk errors):
 - Extraction system definition file
 - Extraction environment definition file
 - Transmission environment definition file
 - Extraction definition file
 - Extraction definition preprocess file
 - Extraction information queue file
 - Extraction master status file
 - Extraction server status file
 - Data linkage file

- Duplexing control file
- HiRDB system log

(b) Command

To execute the update-SQL output facility, execute the `hdestart` command with the `-s -L` options specified. For details about the command, see the `hdestart` command.

(c) Process to be used

When you use the update-SQL output facility, the update-SQL output process starts. The following describes the details of the update-SQL output process:

Process name	Start method	Termination method
hdesqlput	hdestart -s -L	hdestop

The actions of the update-SQL output process are the same as of the transmission process except for the following:

- The update-SQL output process does not establish connection with or send data to the target Datareplicator.
- The update-SQL output process does not update the extraction server status file.
- The update-SQL output process does not terminate, even if HiRDB normal termination or linkage stop log information (`pdrlplstop`) is detected.
- The update-SQL output process does not terminate even when `hdeevent -n 0` is executed.
- The update-SQL output process is run with the `nodemst` method, even when `sendmst` is specified in the `sendcontrol` operand in the extraction system definition.
- When a target system is in the reduced mode, the reduced-mode operation continues even if `false` is specified in the `overwrite_continue` operand in the transmission environment definition.

(2) Update-SQL file

This subsection describes the *update-SQL file* that is output by the update-SQL output facility.

(a) Output target

The update-SQL file is output to the following destination:

Item	Output target
Output target node	Each node at which the source database is to be updated

Item	Output target
Output target directory	Source Datareplicator directory
File name	SQLTXT_server-name_transmission-target-identifier

If any existing file has this name, the information is added to that file.

If there is no untransmitted information in the extraction information queue file, no update-SQL file is created.

(b) Output format

One row of untransmitted information is output to the update-SQL file as one SQL statement. The data output as SQL statements includes only the transactions that were committed; no data is output for any transaction that was rolled back. The output transactions are sorted in the order they were committed. Update information for which processing is underway is not output.

The following shows the format of the data that is output:

INSERT statement

```
/* update-date */ INSERT INTO authorization-identifier.table-name (column-name , column-name , ... )
VALUES (value#1 , value , ... ) ;
```

UPDATE statement^{#2}

```
/* update-date */ UPDATE authorization-identifier.table-name SET3
column-name=value#1 , column-name=value , ... WHERE#4 column-name=value AND column-name=value
... ;
```

DELETE statement

```
/* update-date */ DELETE FROM authorization-identifier.table-name WHERE#4 column-name=value#1 AND
column-name=value ... ;
```

PURGE TABLE statement^{#5}

```
/* update-date */ PURGE TABLE authorization-identifier.table-name ;
```

Event^{#6}

```
/* ****_**_** **:*:*:* */ /* HDEEVENT event-code */
```

End of transaction

```
/* ****_**_** **:**:** */ COMMIT;
```

#1

The table below shows the format and contents of the column values that are output. For repetition columns, *MCOL* is output as the column value regardless of whether the value is the null value or a non-null value.

Column attribute	Character set	-H option in the hdestart command	Output format	Output value	Remarks
CHAR OR VARCHAR	Omitted	Omitted	Character format (Example: 'aaa')	Value in the corresponding column	A control code is output as a period.
		Specified	Hexadecimal format (Example: x'C3C3C3')		--
	Specified	Omitted	Character format (Example: 'aaa')	Value obtained by converting the character codes of the corresponding column's value to the database locale	A control code is output as a period.
		Specified	Hexadecimal format (Example: x'C3C3C3')		--
MCHAR OR MVARCHAR	--	Omitted	Character format (Example: M'あいうえお')	Value in the corresponding column	A control code is output as a period.
		Specified	Hexadecimal format (Example: x'81C141')		--
NCHAR OR NVARCHAR	--	Omitted	Character format (Example: N'あうお')		A control code is output as a period.
		Specified	Hexadecimal format (Example: x'81C181C2')		--
INTEGER OR SMALLINT	--	--	Decimal format (Example: 100)		--

9. Error Handling Procedures

Column attribute	Character set	-H option in the hdestart command	Output format	Output value	Remarks
DECIMAL	--	--	Decimal format (Example: 100)		If an error occurs while data in packed format is being converted to a character string, that column is output in hexadecimal format.
FLOAT OR SMALLFLT	--	--	Floating-point number format (Example: 1.0...0E+02)		--
DATE	--	--	Character format (Example: '2008-04-15')		If an error occurs while data in packed format is being converted to a character string, that column is output in hexadecimal format.
TIME	--	--	Character format (Example: '16:15:30')		
TIMESTAMP	--	--	Character format (Example: '2008-04-15 16:15:30')		
INTERVAL YEAR TO DAY	--	--	Decimal format (Example: +12340102.)		
INTERVAL HOUR TO SECOND	--	--	Decimal format (Example: +112233.)		

Column attribute	Character set	-H option in the hdestart command	Output format	Output value	Remarks
BLOB	--	--	Fixed character string	"*BLOB*"	<ul style="list-style-type: none"> The corresponding fixed character string ("*BLOB*" or "*BINARY*") is output also for SUBSTR operation. The corresponding fixed character string ("*BLOB*" or "*BINARY*") is output regardless of whether the value is the null or a non-null value.
BINARY	--	--		"*BINARY*"	
ADT	--	--	Fixed character string	"*ADT*"	The corresponding fixed character string "*ADT*" is output regardless of whether the value is the null value or a non-null value.

Legend:

--: Not applicable

#2

When update information for repetition columns is output, the following limitations apply:

- UPDATE ADD, UPDATE SET, and UPDATE DELETE are all output as UPDATE SET.
- Even for update processing with element specification, neither subscript for column name nor asterisk (*) is output.

#3

If variable-length data with a defined length of 256 bytes or more has not been updated, that column is not output to the SET clause.

#4

The mapping key column is output to the WHERE clause. If the column value is NULL, the output format is *column-name* IS NULL.

#5

If update information for PURGE TABLE is detected for a partitioned table spanning multiple servers, the operation depends on whether the prg_eventno operand is specified in the transmission environment definition. The following table describes the operation depending on the specification value:

Operand specification	Operation
Specified	The facility outputs the event with the number specified in the prg_eventno operand, and then resumes processing.
Omitted	The facility ignores the corresponding PURGE TABLE.

#6

Although event code 0 is not sent to the target Datareplicator, it is output to this file.

If multiple events are issued within a single transaction, only the last event that was issued is output. If the event was issued by the hdeevent command, COMMIT is output immediately after output of the event.

(c) Output example

The following shows an output example of an update-SQL file:

```

/* 2004-12-29 19:18:00 */ INSERT INTO "USR1"."T1" ("C1","C2") VALUES(1,'a');
/* 2004-12-29 19:18:30 */ UPDATE "USR1"."T1" SET "C1"=2,"C2"='b' WHERE "C1"=2;
/* 2004-12-29 19:19:00 */ DELETE FROM "USR1"."T1" WHERE "C1" IS NULL;
/* 2004-12-29 19:19:30 */ PURGE TABLE "USR1"."T1";
/* ****_*** **:*:* */ /* HDEEVENT 200 */
/* ****_*** **:*:* */ /* COMMIT;
    
```

(3) Procedure

To execute update-SQL output:

1. Terminate the source Datareplicator, if it is active.

This prevents connection from being established with the target system if an invalid option is specified in step 3 or 6.

```
hdsstop -t immediate
```

2. If `true` was specified in the `overwrite` operand in the transmission environment definition, terminate the source Datareplicator and change the `overwrite` operand to `false`.

This prevents the operation from being switched to reduced mode if the extraction information queue file becomes full in step 3.

3. Start the source Datareplicator's extraction processing.

```
hdestart -e
```

4. Use the following method to check that all update information has been extracted from the system log information:

- Make sure that the source Datareplicator's error information file does not contain any `KFRB00042-E` messages (extraction information queue file is full).

If a `KFRB00042-E` message has been issued, add an extraction information queue file, and then restart extraction processing.[#]

```
hdemodq
```

#

There can be a maximum of 16 extraction information queue files. If the extraction information queue file becomes full while there are already 16 extraction information queue files, some update information has not been output.

- Check the extraction status in the HiRDB system log information to make sure that extraction of all required update information has been completed.

Compare `System Log Extract Point` in the execution results of the `pdlogls` and `pdl` `-d rpl -j` commands to make sure that the latter has passed the former. The following shows an example:

Output example of `pdlogls`

```
$ pdlogls -d sys -s flora370
HOSTNAME : flora370(151739)
Group    Type Server  Gen No.  Status  Run ID      Block No.
log10    sys  sds01      1  oc-d--u  3e6835a9      1      40
log11    sys  sds01      0  os----- 00000000      0      0
```

Output example of `pdl`

9. Error Handling Procedures

```
$ pdls -d rpl -j -s flora370
SYSTEMID      : HRD1(150621)
Data replication : Y
UNITID        : unt1(150621)
Data replication : Y
SERVER NAME    : sds01
Extract Database : Y
Extract Status  : C
System Log Extract Point :
Run ID  Group   Gen No.  BLock No.
3e6835a9 log10  1        41
System Log Sync Info :
Run ID  Group   Gen No.  BLock No.
3e6835a9 log10  1        13
```

According to the result of `pdlogls`, the last block number is block 40 in `log10`, while it is block 41 in `log10` with `pdls`. This indicates that extraction of all the update information has been completed.

5. Terminate the source Datareplicator.

```
hdestop
```

6. Delete all update-SQL files, and then start update-SQL output processing on the source Datareplicator.

```
hdestart -s -L
```

7. Execute the `hdestate` command to verify that the values of `Queue write position` and `Queue current pos` are the same.

For details about `Queue current pos`, see the `hdestate` command.

8. Terminate the source Datareplicator.

```
hdestop
```

9. Open the update-SQL file with a program such as a text editor and check the untransmitted data.

If you need to send untransmitted data to the target, start transmission processing. If there is no need to send data, initialize or partially initialize the source Datareplicator in order to delete the untransmitted data stored in the extraction information queue file.

(4) Notes

In the messages displayed by Datareplicator, the transmission and output processes are both indicated as `Sender process`. To determine whether a message was intended for the transmission process or the output process, check the message output process name

that is displayed in the message.

- If system switchover occurs during execution of the update-SQL output facility while `server` is specified in the `nodecontrol` operand in the extraction system definition, Datareplicator performs the processing described in the following table.

Table 9-16: Datareplicator processing when system switchover occurs during execution of the update-SQL output facility

System switchover status	Datareplicator processing
System switchover occurs at the system where MST is located	After system switchover is completed, MST is started, but it rejects a connection request from NMT that is running in the update-SQL output status. As a result, all NMTs and MSTs are terminated.
System switchover occurs at the system where NMT is located	After system switchover is completed, NMT is started, [#] but it rejects a connection request from MST that is running in the update-SQL output status. As a result, the NMT where system switchover occurred is terminated.

Legend:

MST: Extraction master process

NMT: Extraction node master process

#

Because NMT is started by the `hdestart_n` command, both extraction and transmission processes are started.

Chapter

10. Messages

This chapter explains the message output format and the handling of messages.

- 10.1 Overview of messages
- 10.2 Details about messages
- 10.3 List of system call errors
- 10.4 List of cause codes

10.1 Overview of messages

This section explains the messages that are issued by Datareplicator. For details about the messages that are output by HiRDB, see the *HiRDB Version 9 Messages* manual.

10.1.1 Message output destination

This section explains the destinations of Datareplicator messages.

(1) *UNIX (HP-UX, Solaris, AIX, or Linux)*

The UNIX Datareplicator outputs messages to the following destinations:

- Standard error output
- Error information files

Source Datareplicator:

Extraction master error information files and extraction node master error information files

Target Datareplicator:

Import error information files

(2) *Windows*

The Windows Datareplicator outputs messages to the following destinations:

- Standard error output
- Error information files

This is the window where the Datareplicator command was executed.

Source Datareplicator:

Extraction master error information files and extraction node master error information files

Target Datareplicator:

Import error information files

- Event log

The Windows Datareplicator outputs messages to the event log, while the UNIX Datareplicator outputs messages to the syslog file.

10.1.2 Message output format

This section explains the message output format for each output destination.

(1) When the output destination is the standard error output, syslog file, or event log

Datareplicator outputs messages to these destinations in the following format:

KFRBnnnnn-T XX YY (ZZ...ZZ) MM...MM

KFRBnnnnn-T

Message ID (11 alphanumeric characters)

XX

Identifier of the source or target Datareplicator (two hexadecimal characters)

However, in the following cases, . . is displayed in place of the source or target Datareplicator identifier because the actual identifier cannot be displayed:

- Error is output from a command
- Error is output before shared memory has been allocated
- Error is output before message files have been opened

YY

Data linkage identifier (two hexadecimal characters)

However, in the following cases, . . is displayed in place of the data linkage identifier because the actual data linkage identifier cannot be displayed:

- Error is output from a command
- Error is output before shared memory has been allocated
- Error is output before message files have been opened
- Error that is output does not depend on data linkage identifiers

ZZ...ZZ

Name of process and import group from which message is output

An import group name is displayed only in the case of an error resulting from an import process.

MM...MM

Message text (maximum of 242 characters)

(2) When the output destination is an error information file

Datareplicator uses error information files to accumulate logs of errors occurring in the system. For details about the error information file output formats and handling, see *6.4.2 Handling of the files used with the source Datareplicator* for the source Datareplicator and *6.7.2 Handling of the files used with the target Datareplicator* for

the target Datareplicator.

10.1.3 Message descriptive format

This section explains the format used in this manual to explain messages.

(1) Descriptive format

KFRBnnnnn - T

English message text (Y)

Explanation of the message

S: Datareplicator processing that takes place when the message is issued.

O: Action that is to be taken by the operator who receives the message.

[Action]: Action that is to be taken by the Datareplicator administrator who receives the message.

Note:

- *Contact the customer support center* in the message descriptions means that the user is to contact their designated support engineer or the sales department.
- Some messages have only English message text; they cannot be output in Japanese.

(2) Components of the message ID

The message ID consists of the following components:

KFRB

Indicates that the message was issued by Datareplicator.

nnnnn

Message number

T

Message's severity level:

E

Error message: Indicates that an error occurred that prevents execution of a facility.

W

Warning message: Indicates that a warning has been issued with respect to resource utilization status, for example, or that a command specification is invalid, but processing will continue using the default value.

I

Information message: Indicates a report on a simple activity status that is neither E nor W.

Q

Query message: The system is waiting for a response from the user (Datareplicator will wait for the user to enter a response to the output message).

Y

Message output destination (if a message has multiple destinations, all the applicable destinations are listed, delimited by the plus sign (+)):

C

Error information file, and syslog file[#]

S

Standard error output

L

syslog file[#]

E

Error information file or event log (applicable to Windows)

#

In Windows, the message is output to the event log.

(3) Explanation of the message text

(a) Information provided in a message text

The message text explains the message or event, and provides embedded character strings as necessary. The following symbols are used in message texts:

{ }

Only one of the texts enclosed in braces is displayed in the message. The items within the braces are delimited by the vertical bar (|).

Example

{ authorization-identifier | password }

[]

A text enclosed in brackets might not always be displayed.

(b) Selecting English or Japanese messages

Datareplicator's message texts can be provided in either English or Japanese. You select English or Japanese messages by your specification of the `msglocale` operand in the extraction system definition or import system definition.

This selection is available only for messages that are output to the error information files and syslog files. English-only messages are output to the standard output and when language selection is not available; in these cases, Japanese message texts are not provided.

(4) Explanation of detail information

Detail information is provided with some error messages when they are output to the error information files.

(a) Extraction transaction information

The extraction transaction information is input to the data linkage recovery facility and uniquely identifies transactions.

When extraction transactions are not output

When Datareplicator Extension is used for import processing, extraction transaction information is not output.

Message numbers that output extraction transactions as detail information:

The extraction transaction information is output as detail information for the messages with the following numbers:

03004, 03006, 03009, 03010, 03021, 03022, 03023, 03024, 03025, 03027, 03028, 03029, 03031, 03032, 03033, 03034, 03036, 03038, 03040, 03041, 03042, 03043, 03044, 03045, 03046, 03047, 03048, 03049, 03051, 03052, 03053, 03056, 03057, 03058, 03070, 03071, 03074, 03080, 03081, 03083, 03084, 03085, 03086, 03087, 03088, 03089, 03090, 03091, 03092, 03101, 03102, 03103, 03104, 03105, 03106

Output format of extraction transaction information

```
<text> Additional Transaction Info = nn...nn.
```

If the message number is not 03009, the displayed information is about the transaction being processed. If the message number is 03009, the displayed information is about the last transaction that was processed before a synchronization point was issued. `nn...nn` displays the following information:

When the sender is XDM/DS and the source DBMS is not TMS-4V/SP:

Activity ID (in hexadecimal)

When the sender is XDM/DS and the source DBMS is TMS-4V/SP:

APJ journal ID, in the format [*a-*] *bbbbbbbb-ccccccc-ddd*.

a: DMS-ID (not output if it is 0x00)

bbbbbbbb: Run ID (in hexadecimal)

ccccccc: Sequential central processing number (in hexadecimal)

ddd: JSQ (in decimal)

(b) Update information identifier

The update information identifier is specified when the import suppress facility is used in order to uniquely identify update information.

Update information identifier output conditions

The update information identifier is output when the following conditions are satisfied:

- The applicable transmission source uses a version listed in the following table

Table 10-1: Program and version that add detailed information to messages

Source program	Condition	Version
HiRDB Datareplicator	None	04-00-/O or later
HiRDB Datareplicator Version 5.0	None	05-03-/D or later
HiRDB Datareplicator Version 6	None	06-01-/A or later
HiRDB Datareplicator Version 7	None	07-00 or later
HiRDB Datareplicator Version 8	None	08-00 or later
XDM/DS	Extraction from TMS/4V	07-01 or later
	Extraction from other source	08-03 or later
DBDR	None	01-00 or later

- Datareplicator Extension is not used for import processing

Message numbers that output update information identifier as detail information

The update information identifier is output as detail information for the messages with the following numbers:

03004, 03010, 03028, 03029, 03031, 03032, 03033, 03034, 03036, 03038, 03040, 03041, 03042, 03043, 03044, 03045, 03046, 03047, 03048, 03049, 03051, 03052, 03053, 03056, 03057, 03058, 03070, 03071, 03074, 03080, 03081, 03083, 03084,

10. Messages

03085, 03086, 03087, 03088, 03089, 03090, 03091, 03092, 03101, 03102, 03103,
03104, 03105, 03106

Output format of update information identifier

```
<text> ExtractID = nn...nn.
```

10.2 Details about messages

KFRB00001-E

File(pipe) open error, file name = *aa...aa*, errno = *xx...xx*. (C)

An open error occurred on a file or pipe.

aa...aa: Filename

One of the following keywords might be displayed instead of the actual file name:

- `StatusFile`: Extraction server status file
- `RECEIVING QUEUE FILE`: Import information queue file
- `STATUS FILE`: Import status file
- `ERRORLOG FILE`: Error information file
- `UNREFLECTED DATA FILE`: Unimported information file
- `START UP PARAMETER FILE`: Import system definition
- `REFLECT ENVIRONMENT FILE`: Import environment definition
- `REFLECT DEFINITION FILE`: Import definition
- `PIPE`: Process-to-process communication PIPE
- `SOCKET`: Communication socket

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00002-E

File(pipe) read error, file name = *aa...aa*, errno = *xx...xx*. (C)

A read error occurred on a file or pipe.

aa...aa: Filename

One of the following keywords might be displayed instead of the actual file name:

- `StatusFile`: Extraction server status file
- `RECEIVING QUEUE FILE`: Import information queue file
- `STATUS FILE`: Import status file
- `ERRORLOG FILE`: Error information file

- UNREFLECTED DATA FILE: Unimported information file
- START UP PARAMETER FILE: Import system definition
- REFLECT ENVIRONMENT FILE: Import environment definition
- REFLECT DEFINITION FILE: Import definition
- PIPE: Process-to-process communication PIPE
- SOCKET: Communication socket

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation. If the error number is 0, the contents of the file are invalid.

KFRB00003-E

`File(pipe) write error, file name = aa...aa, errno = xx...xx. (C)`

A write error occurred on a file or pipe.

aa...aa: Filename

One of the following keywords might be displayed instead of the actual file name:

- `StatusFile`: Extraction server status file
- RECEIVING QUEUE FILE: Import information queue file
- STATUS FILE: Import status file
- ERRORLOG FILE: Error information file
- UNREFLECTED DATA FILE: Unimported information file
- START UP PARAMETER FILE: Import system definition
- REFLECT ENVIRONMENT FILE: Import environment definition
- REFLECT DEFINITION FILE: Import definition
- PIPE: Process-to-process communication PIPE
- SOCKET: Communication socket

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation. If the error number is 0, the disk space is insufficient.

KFRB00004-E

File(pipe) close error, file name = *aa...aa*, errno = *xx...xx*. (C)

A close error occurred on a file or pipe.

aa...aa: Filename

One of the following keywords might be displayed instead of the actual file name:

- StatusFile: Extraction server status file
- RECEIVING QUEUE FILE: Import information queue file
- STATUS FILE: Import status file
- ERRORLOG FILE: Error information file
- UNREFLECTED DATA FILE: Unimported information file
- START UP PARAMETER FILE: Import system definition
- REFLECT ENVIRONMENT FILE: Import environment definition
- REFLECT DEFINITION FILE: Import definition
- PIPE: Process-to-process communication PIPE
- SOCKET: Communication socket

xx...xx: Error number set in errno

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00005-E

Unable to find update queue file. (C)

Datareplicator cannot write the update data because there is no available import information queue file.

S: Cancels reception processing.

O: See *Insufficient file space - Import information queue file* in Table 9-3 *Causes of errors and handling methods*.

KFRB00006-E

Insufficient memory, size = *aa...aa*, info = *bb...bb*. (C)

A memory shortage occurred.

aa...aa: Requested size (-1 if internal system call processing resulted in a memory shortage)

bb...bb: Additional information

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB00007-E

Failed to generate process, code = *xx*. (C)

Process generation failed.

xx: Detail code indicating the process that was to be generated:

00: Import master process

01: Import communication master process

02: Reception process

03: Import definition server process

04: Import process

05: Import SQL process

S: Cancels processing.

O: Re-execute. If the same error recurs, contact the customer support center.

KFRB00008-E

Parent process disappeared, process name = *aa...aa*. (C)

The parent process disappeared for some reason.

aa...aa: Name of the process that disappeared

S: Cancels processing.

O: Re-execute. If Datareplicator does not function normally, re-execute using the `hdsstart -i` command.

KFRB00009-E

Internal error, function name = *aa...aa*, code = *xx*. (S+L+E)

Internal conflict occurred.

aa...aa: Name of the function resulting in the internal conflict

xx: Detail code

S: Cancels processing.

O: Contact the customer support center.

KFRB00010-E

Read/write pointer set error (`lseek`), file name = *aa...aa*, `errno` = *xx...xx*. (C)

A read or write pointer set error occurred (`seek`).

aa...aa: Filename

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00011-I

No log data in update queue file. (C)

The import information queue file contains no data.

S: Resumes processing.

KFRB00012-E

Update queue file is full. (C)

There is no space in the import information queue file.

S: Cancels reception processing.

O: After import processing is terminated, specify an import information queue file with enough space to accumulate data, and then restart Datareplicator with the `hdsstart -i` command. For details about determining the size of an import information queue file, see Chapter 4. *System Design*.

KFRB00013-W

Syslog file write error. (C)

A write error occurred in the syslog file.

S: Resumes processing. However, Datareplicator does not output messages to the syslog file.

KFRB00014-W

Reflect information file write error, function name = *aa...aa*, `errno` = *xx...xx*. (C)

A write error occurred in the import error information file.

aa...aa: Name of the function resulting in the error

xx...xx: Error number set in `errno`

S: Resumes processing. However, Datareplicator does not output messages to the import error information file.

O: Check the displayed message, eliminate the cause of the error, and then re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00015-W

Unreflected data file write error, function name = *aa...aa*, `errno` = *xx...xx*. (C)

A write error occurred in the unimported information file.

aa...aa: Name of the function resulting in the error

xx...xx: Error number set in `errno`

S: Resumes processing. However, Datareplicator does not output unimported information to the file.

O: Check the displayed message, eliminate the cause of the error, and then re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00016-W

Reflect information /unreflected data file initialize error, function name = *aa...aa*, `errno` = *xx...xx*. (C)

An initialization error occurred in the import error information file or unimported information file.

aa...aa: Name of the function resulting in the error

xx...xx: Error number set in `errno`

S: Resumes processing. However, Datareplicator does not output information to the import error information file or unimported information file.

O: Check the displayed message, eliminate the cause of the error, and then re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00017-W

Reflect information /unreflected data file open error, function name = *aa...aa*, `errno` = *xx...xx*. (C)

An open error occurred in the import error information file or unimported information file.

aa...aa: Name of the function resulting in the error

xx...xx: Error number set in `errno`

S: If an open error occurs in the import error information file, Datareplicator resumes processing. If an open error occurs in the unimported information file, Datareplicator cancels processing. Datareplicator does not output information to the import error information file or unimported information file.

O: Check the displayed message, eliminate the cause of the error, and then re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00018-E

Exec error, program = *aa...aa*, `errno` = *xx...xx*. (C)

A loading error occurred in a program.

aa...aa: Name of the program

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00019-E

Fork error, process = *aa...aa*, `errno` = *xx...xx*. (C)

Process generation failed.

aa...aa: Process name

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00020-E

Malloc error, `errno` = *xx...xx*, size = *aa...aa*, info = *bb...bb*. (C)

A memory shortage occurred while allocating an area.

xx...xx: Error number set in `errno`

aa...aa: Requested size

bb...bb: Detail information

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00021-E

Semaphore delete error, `errno` = *xx...xx*. (C)

Semaphore deletion processing failed.

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00022-E

Semaphore id get error, `errno` = *xx...xx*, Key = *aa...aa*. (C)

Semaphore allocation failed.

xx...xx: Error number set in `errno`

aa...aa: Semaphore key

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00023-E

Semaphore lock or unlock error, `errno = xx...xx`. (C)

Semaphore lock or unlock processing failed.

`xx...xx`: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00024-E

File information get error (`stat`), `errno = xx...xx`. (C+S)

Acquisition of file status information failed.

`xx...xx`: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00025-E

Unable to attach shared memory, shared memory type = `aa...aa`, `errno = xx...xx`, ID = `bb...bb`. (C)

Shared memory attach processing failed.

`aa...aa`: Type of shared memory:

COMMON (shared memory common to import processing)

DEFSEV (shared memory for analyzing import definitions)

ERRTXT (shared memory for message text)

EXTCOM (shared memory common to extraction processing)

EXTDEF (shared memory for analyzing extraction definitions)

LOBSHM (shared memory for importing BLOB data)

NMTCMD (shared memory for communication between extraction node master and command)

STATINF (shared memory for status information)

SYNCINF (shared memory for synchronous import group)

TBILIST (shared memory for transaction branch information management list)

xx...xx: Error number set in `errno`

bb...bb: Shared memory ID

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00026-E

Unreflected data file is full. (C)

The unimported information file is full.

S: Cancels processing.

O: Increase the `unreffilesz` operand value in the import environment definition, and then restart import processing.

KFRB00027-E

Unable to establish language environment. (C)

Setup of the character code set environment failed.

S: Cancels processing.

O: If sufficient memory is allocated to read the code mapping table file, but this message is issued, re-create the code mapping table.

KFRB00028-E

Unable to set up signal handler, `errno = xx...xx`. (C)

Registration of a signal handler failed.

xx...xx: Error number set in `errno`

S: Terminates the system.

O: This is an internal error. Contact the customer support center.

KFRB00029-E

Unable to allocate shared memory, shared memory type = *aa...aa*, `errno = xx...xx`, Key = *bb...bb*, size = *cc...cc*. (C)

An allocation error occurred in the shared memory.

aa...aa: Type of shared memory:

COMMON (shared memory common to import processing)

DEFSERV (shared memory for analyzing import definitions)

ERRTXT (shared memory for message text)

EXTCOM (shared memory common to extraction processing)

EXTDEF (shared memory for analyzing extraction definitions)

LOBSHM (shared memory for importing BLOB data)

NMTCMD (shared memory for communication between extraction node master and command)

STATINF (shared memory for status information)

SYNCINF (shared memory for synchronous import group)

TBILIST (shared memory for transaction branch information management list)

TRNINF (shared memory for transactions management)

xx...xx: Error number set in `errno`

bb...bb: Shared memory key

cc...cc: Requested size

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00030-E

No error message data in message text file, message id = *nn...nn*.
(C)

There is no message text in the message file.

nn...nn: Message ID

S: Resumes processing.

O: This is an internal error. Contact the customer support center.

KFRB00031-E

Update queue file sequence error. (C)

The sequence of the import information queue files is invalid.

The sequence or contents of the import information queue files is invalid.

S: Cancels processing.

O: This is an internal error. Contact the customer support center.

KFRB00032-E

Error occurred in getting file status, file name = *aa...aa*, `errno` = *xx...xx*. (C)

An error occurred while obtaining a file status.

aa...aa: Filename

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00033-E

Invalid transmission queue file, file name = *aa...aa*. (C+S)

The contents of the extraction information queue file are invalid.

aa...aa: Filename

S: Cancels processing.

O: This is an internal error. Contact the customer support center.

KFRB00034-E

Invalid extract status file, file name = *aa...aa*. (C+S)

The contents of the status file are invalid during extraction processing.

aa...aa: Filename

S: Cancels processing.

O: This is an internal error. Contact the customer support center.

KFRB00035-E

Error occurred in getting file-disk information, file name = *aa...aa*. (C)

Acquisition of file disk information failed.

aa...aa: Filename

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00036-W

Extract master information file initialization error, function name = *aa...aa*, `errno` = *xx...xx*. (C)

An initialization error occurred in the extraction master error information file.

aa...aa: Function name

xx...xx: Error number set in `errno`

S: Resumes processing. However, Datareplicator does not output information to the extraction master error information file.

O: Check the displayed message, eliminate the cause of the error, and then re-execute.

For details about the error number, see `errno.h` or the OS documentation.

KFRB00037-W

Extract slave information file initialization error, function name = *aa...aa*, `errno` = *xx...xx*. (C)

An initialization error occurred in the extraction node master error information file.

aa...aa: Function name

xx...xx: Error number set in `errno`

S: Resumes processing. However, Datareplicator does not output information to the extraction node master error information file.

O: Check the displayed message, eliminate the cause of the error, and then re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00038-W

Extract master information file open error, function name = *aa...aa*, `errno` = *xx...xx*. (C)

An open error occurred in the extraction master error information file.

aa...aa: Function name

xx...xx: Error number set in `errno`

S: Resumes processing. However, Datareplicator does not output information to the extraction master error information file.

O: Check the displayed message, eliminate the cause of the error, and then re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00039-W

Extract slave information file open error, function name = *aa...aa*, `errno` = *xx...xx*. (C)

An open error occurred in the extraction node master error information file.

aa...aa: Function name

xx...xx: Error number set in `errno`

S: Resumes processing. However, Datareplicator does not output information to the extraction node master error information file.

O: Check the displayed message, eliminate the cause of the error, and then re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00040-W

Extract master information file write error, function name = *aa...aa*, `errno` = *xx...xx*. (C)

A write error occurred in the extraction master error information file.

aa...aa: Function name

xx...xx: Error number set in `errno`

S: Resumes processing. However, Datareplicator does not output information to the extraction master error information file.

O: Check the displayed message, eliminate the cause of the error, and then re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00041-W

Extract slave information file write error, function name = *aa...aa*, `errno` = *xx...xx*. (C)

A write error occurred in the extraction node master error information file.

aa...aa: Function name

xx...xx: Error number set in `errno`

S: Resumes processing. However, Datareplicator does not output information to the extraction node master error information file.

O: Check the displayed message, eliminate the cause of the error, and then re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00042-E

Transmission queue file is full. (C)

The extraction information queue file is full.

S: Retries accumulation processing at the specified interval until enough space becomes available.

O: If transmission processing has stopped, restart it.

KFRB00043-I

No log data in transmission queue file. (C)

The extraction information queue file contains no update information.

S: Resumes processing.

KFRB00044-E

Unable to set up signal handler, kind = *aa...aa*, `errno` = *xx...xx*. (C)

Registration of signal handler failed.

aa...aa: Signal type

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00045-E

Invalid extract master status file. (C)

The contents of the extraction master status file are invalid.

S: Cancels processing.

O: This is an internal error. Contact the customer support center.

KFRB00046-E

Ftok error, file name = *aa...aa*, errno = *xx...xx*. (C+S)

A key creation error occurred in a file.

aa...aa: Filename

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00047-E

File lock/unlock error, file name = *aa...aa*, errno = *xx...xx*. (C+S)

File lock/unlock processing failed.

aa...aa: Filename

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00048-E

Process terminated due to node-master abnormal termination. (C)

The process terminated because the extraction node master process terminated abnormally.

S: Cancels processing.

O: Check the cause of the error in the extraction node master process.

KFRB00049-E

Invalid extract coordinate file. (C)

The data linkage file is invalid.

S: Cancels processing.

O: This is an internal error. Contact the customer support center.

KFRB00050-E

Extract data exceeds 32000byte, table id = *aa...aa*. (C+S)

Update information exceeded 32,000 bytes in length. When the target Datareplicator version is earlier than 02-00, update information exceeding 32,000 bytes cannot be transmitted.

S: Cancels processing.

O: Correct the extraction definitions or other source of the problem so that the update information does not exceed 32,000 bytes. If the error occurred during execution of the `hdeprep` command, correct the error, and then re-execute. If the error occurred during the transmission process, synchronize the data linkage environments at the source and target, initialize them, and then re-create the target database on the basis of the source database before restarting HiRDB Datareplicator linkage.

KFRB00051-I

Error information/unreflected data file swapped, file kind = *aa...aa*, closed file = *bb...bb*. (L)

Error information files or unimported information files were swapped.

aa...aa: Type of files that were swapped:

`errorfile`: Import error information files or extraction node error information files

`unreffile`: Import information files at the target

`msterrrrfile`: Extraction master error information files

bb...bb: Name of the file that was closed as a result of being swapped out

The Datareplicator identifier and data linkage identifier are displayed as ...

S: Resumes processing.

O: Back up the file, if necessary.

KFRB00052-I

Error information/unreflected data file closed, file kind = *aa...aa*, closed file = *bb...bb*. (L)

The error information file or unimported information file was closed.

aa...aa: Type of file that was closed:

`errorfile`: Import error information file or extraction node error information file

`unreffile`: Import information file at the target

`msterrfile`: Extraction master error information file

bb...bb: Name of the file that was closed

The Datareplicator identifier and data linkage identifier are displayed as . . .

S: Resumes processing.

O: Back up the file, if necessary.

KFRB00060-E

Error occurred in Oracle Call Interface function, func = *aa...aa*,
return = *bb...bb*, errcode = *cc...cc*, info = *dd...dd*. (C+S)

An error occurred in the OCI function.

aa...aa: Name of the OCI function

bb...bb: Function's return value

cc...cc: Oracle error code

dd...dd: Additional information

S: Terminates processing.

O: Check the Oracle error code, eliminate the cause of the error, and then re-execute.
If the Oracle error code is 1422, there might be multiple event control tables
(hde_dtbl) in the source Oracle. Delete any extra event control tables so that there is
only one event control table in the source Oracle.

KFRB00061-E

Unmatch extract DBMS kind, initial = *aa...aa*, current = *bb...bb*.
(C+S)

There is no match with the DBMS that is subject to extraction processing.

aa...aa: DBMS type during initialization:

0: HiRDB

1: Oracle

2: SQL Server

bb...bb: DBMS type during execution:

0: HiRDB

1: Oracle

2: SQL Server

S: Terminates processing.

O: Re-execute with the command for the DBMS used during initialization.

KFRB00062-E

Failed to load library, library name = *aa...aa*. (E)

Loading of a library failed. Library name = *aa...aa*

aa...aa: Library name

S: Terminates processing.

O: Eliminate the cause of the error on the basis of the error information displayed as *information*. *information* can display a maximum of 254 bytes of characters. If the error information exceeds this length, only the last 254 bytes of characters are displayed.

KFRB00063-E

Failed to get procedure address, library name = *aa...aa*, procedure name = *bb...bb*. (E)

Loading of a library failed.

Acquisition of function address failed.

aa...aa: Library name

bb...bb: Function name

S: Terminates processing.

O: Eliminate the cause of the error on the basis of the error information displayed as *information*. *information* can display a maximum of 254 bytes of characters. If the error information exceeds this length, only the last 254 bytes of characters are displayed.

KFRB00064-E

Operation directory access error occurred, operation = *nn...nn*, directory = *aa...aa*, errno = *xx...xx*. (S+L)

The operation directory cannot be accessed.

nn...nn: Type of operation (system call name)

aa...aa: Operation directory name

xx...xx: Error number

S: Terminates processing.

O: Take the following action:

- When the error number is 2
 - Check and, if necessary, revise the directory name set in the `HDEPATH` or `HDSPATH` environment variable.
 - When the error number is 13
 - Check and, if necessary, revise the access privilege for the operation directory.
- For any other error number, see `errno.h` or the OS documentation.

KFRB00065-E

Shared memory operation failure, type = *aa...aa*, ID = *bb...bb*,
operation = *cc...cc*, errno = *xx...xx*. (C+L)

Manipulation of shared memory failed.

aa...aa: Type of shared memory:

COMMON (shared memory common to import processing)

DEFSEV (shared memory for import definition analysis)

EXTCOM (shared memory common to extraction processing)

EXTDEF (shared memory for extraction definition analysis)

ERRTXT (shared memory for message text)

LOBSHM (shared memory for importing BLOB)

NMTCMD (shared memory for communication between extraction node master and
command)

STATINF (shared memory for status information)

bb...bb: Shared memory ID

cc...cc: Type of operation

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Check `errno.h` or the OS documentation for the error number, eliminate the cause
of the error, and then re-execute.

KFRB00066-W

Delay time exceeded definition value of *aa...aa*, *bb...bb*. (C+E)

The delay period exceeded the defined value.

aa...aa: Defined operand

`extract_delay_limit_time`: Extraction delay period

`send_delay_limit_time`: Transmission delay period

`reflect_delay_limit_time`: Import delay period

bb...bb: Delay period

Displays the delay period in the format

`DelayTime[xxx(hour):xx(minute):xx(second)]`.

S: Resumes processing.

O: Take appropriate action on the basis of the information displayed in *aa...a*:

- When `extract_delay_limit_time` (extraction delay period) is displayed
If the delay period is too long, extend the system log I/O buffer or the extraction information queue I/O buffer for extraction processing.
- When `send_delay_limit_time` (transmission delay period) is displayed
If the delay period is too long, extend the extraction information queue I/O buffer for extraction processing or for transmission processing.
- When `reflect_delay_limit_time` (import delay period) is displayed
If the delay period is too long, compress the amount of update data at the source Datareplicator.

KFRB00067-W

Delay time returned into the definition value of `aa...aa`. (C+E)

The delay time was returned to within the defined value `aa...aa`.

`aa...aa`: Defined operand:

`extract_delay_limit_time`: Extraction delay period
`send_delay_limit_time`: Transmission delay period
`reflect_delay_limit_time`: Import delay period

S: Resumes processing.

O: Take appropriate action according to the KFRB00066-W message.

KFRB00068-E

Process check operation failure, operation = `aa...aa`, errno = `bb...bb`, process = `cc...cc`, pid = `dd...dd`. (C)

Checking for existence of a process failed.

`aa...aa`: Type of operation:

`waitpid`
`kill`

`bb...bb`: Error number set in `errno`

`cc...cc`: Process name

`dd...dd`: Process ID

S: Resumes processing, assuming that the checked process is inactive.

KFRB00069-E

File size exceeded limit, file name = `aa...aa`, code = `bb`. (S+L)

The specified file size exceeded the maximum value for the file system

aa...aa: File name

bb: Detail code:

1: The specified file size is 2 GB or greater, but the file system is not a large file system.

2: The maximum file size was exceeded.

S: Cancels processing.

O: Take appropriate action according to the detail code:

- Detail code is 1

Use an OS system management command to determine whether the file system for the specified file supports large files. If it does not support large files, either set it to support large files or set the maximum file size to less than 2 GB.

For details about the OS system management commands, see the OS documentation.

- Detail code is 2

Set the file size so that it does not exceed the maximum value set for the file system. The permitted maximum value depends on the OS. See *6.11 Handling of large files*.

KFRB00071-W

Message classification set error, information = *aa...aa*, code = *xx...xx*. (C)

Setting of a message type failed.

aa...aa: Maintenance information 1

xx...xx: Maintenance information 2

Note:

This message is output when a file under the installation directory that is created during installation has become corrupted.

S: Assumes HiRDB Datareplicator as the program ID in the system log or the source name in the event log, and then resumes processing.

O: Reinstall HiRDB Datareplicator.

KFRB00080-W

Recover information file access error, function name = *aa...aa*, file name = *bb...bb*, errno = *xx...xx*. (C)

A recovery information file access error occurred.

aa...aa: Name of function resulting in an error

bb...bb: File name

xx...xx: Error number set in `errno`

S: Resumes processing. Note that the recovery information is not output.

O: Check the displayed message, eliminate the cause of the error, and then re-execute.

For details about the error number, see `errno.h` or the OS documentation.

KFRB00081-W

Semaphore operation failure, function name = *aa...aa*, sem-id = *bb...bb*, `errno` = *xx...xx*. (C)

Lock processing failed.

aa...aa: Name of function resulting in an error

bb...bb: Semaphore ID

xx...xx: Error number set in `errno`

S: Resumes processing. Note that the recovery information is not output.

O: Check the displayed message, eliminate the cause of the error, and then re-execute the command.

For details about the error number, see `errno.h` or the OS documentation.

KFRB00082-W

Memory allocate error, function name = *aa...aa*, size = *bb...bb*, `errno` = *xx...xx*. (C)

A memory shortage occurred while allocating an area.

aa...aa: Name of function resulting in an error

bb...bb: Size of area to be allocated

xx...xx: Error number set in `errno`

S: Resumes processing. Note that the recovery information is not output.

O: Check the displayed message, eliminate the cause of the error, and then re-execute.

For details about the error number, see `errno.h` or the OS documentation.

KFRB00083-E

Unable to load compression library, name = *aa...aa*, return code = *bb...bb*, `errno` = *cc...cc*. (C)

A compression library load error occurred.

aa...aa: Compression library name

- PDZLIB

bb...bb: Return code

4: Library *aa...aa* specified by its compression library name during table definition has not been installed in HiRDB.

8: An error occurred while loading compression library *aa...aa*.

12: The compression library was loaded successfully, but a symbol resolution error occurred.

cc...cc: Error number (*errno*)

When *bb...bb* is 4: ****

When *bb...bb* is 8: *errno* during the compression library load processing

When *bb...bb* is 12: *errno* during the compression library symbol resolution processing

S: Cancels processing.

O: Check the return code and error number, eliminate the cause of the error, and then re-execute the command.

For details about the error number, see `errno.h` and the OS documentation. If the cause of the error cannot be eliminated, contact the customer support center.

KFRB00084-E

aa...aa return with invalid code *bb...bb*. (C)

An invalid return value *bb...bb* was returned from compression library *aa...aa*.

aa...aa: Compression library name

- PDZLIB

bb...bb: Return value

Value returned from compression library *aa...aa*

S: Cancels processing.

O: Contact the customer support center.

KFRB00085-E

Length of Expand data *aa...aa* *bb...bb*, compression library name = *cc...cc*. (C)

An error occurred in the compression library. The message text (Length of Expand data *aa...aa* *bb...bb*, compression library name = *cc...cc*) constitutes error information.

aa...aa

- Length of expanded data

- larger than
- Expanded data length less than compressed data length

bb...bb

- Invalid
- Size of expanded data storage area
- Length of compressed data

cc...cc: Compression library name

- PDZLIB

S: Cancels processing.

O: Contact the customer support center.

KFRB00086-E

Expand processing error occurred, library = *aa...aa*, *errinf* = *bb...bb*.
(C)

The error indicated by *bb...bb* occurred while expanding compression library *aa...aa*.

aa...aa: Compression library name

- PDZLIB

bb...bb: Error information

Information about the error that occurred while expanding compression library
aa...aa

S: Cancels processing.

O: Eliminate the cause of the error based on the error information shown below, and then re-execute the command. If the cause of the error cannot be eliminated, contact the customer support center.

- Value less than -1,000 (*-1nnn*)

A value less than -1,000: Reference `errno.h` and the OS documentation by using `-(error information + 1,000)` (*nnn*) as the value of `errno` (1 to 151), eliminate the cause of the error, and then re-execute the command.

For the typical values of `errno`, see the manual *HiRDB Version 9 Messages*.

- -4

Add about 260 KB of memory required for executing the compression library, and then re-execute the command.

- Other

Contact the customer support center.

KFRB00100-I

Target site Datareplicator start completed. (C)

The target Datareplicator has started.

S: Resumes processing.

KFRB00101-I

Reflection restart event was accepted. (C)

An import processing restart event has been accepted.

S: Resumes processing.

KFRB00102-I

Command request was accepted. (C)

A command request has been accepted.

S: Resumes processing.

KFRB00103-I

Accepted target site Datareplicator stop request. Requiring child processes to terminate. (C)

The target Datareplicator termination request has been accepted. The termination request will be issued to subprocesses.

S: Resumes processing.

KFRB00104-I

Target site Datareplicator termination completed. (C)

Target Datareplicator termination processing has been completed.

S: Terminates the target Datareplicator.

KFRB00150-E

Target site Datareplicator start error. (C)

Startup of the target Datareplicator failed.

S: Terminates the target Datareplicator.

O: Eliminate the cause of the error and restart the target Datareplicator.

KFRB00151-E

Illegal replication node-id was specified. (C)

An invalid data linkage identifier was specified.

S: Ignores the command.

O: Re-execute the command with the correct data linkage identifier specified.

KFRB00152-E

Unable to restore communication master process status, protocol = *aa...aa*. (C)

Datareplicator is unable to restore the status of the import communication master process.

aa...aa: Protocol type

S: Skips startup of the corresponding import command master process and resumes processing.

O: Correct the error so that Datareplicator can read the recovery information from the import status file, and then restart the system. In the event of a medium error, initialize the medium again.

KFRB00153-E

Communication master process not exist, protocol = *aa...aa*. (C)

There is no import communication master process.

aa...aa: Protocol type

S: Restarts the import communication master process.

O: If restart fails, eliminate the cause of the error so that Datareplicator can start the import communication master process, and then restart the target Datareplicator.

KFRB00154-E

Definition server process not exist. (C)

There is no import definition server process.

S: Resumes processing.

O: Restart import processing.

KFRB00155-E

Communication master process retry started, protocol = *aa...aa*. (C)

Restart of the import communication master process has begun.

aa...aa: Protocol type

S: Resumes processing.

KFRB00156-E

Communication master process retry error, protocol = *aa...aa*. (C)

Restart of the import communication master process failed.

aa...aa: Protocol type

S: Resumes processing.

O: Eliminate the cause of the error so that Datareplicator can start the import

communication master process, and then restart the target Datareplicator.

KFRB00157-E

Reflection restart event acceptance error. (C)

An acceptance error occurred with respect to an import processing restart event.

S: Ignores the event.

O: This is an internal error. Check the error message displayed immediately before this message. If no error message was displayed, contact the customer support center.

KFRB00158-E

Command acceptance error. (C)

An acceptance error occurred with respect to a command.

S: Ignores the command.

O: This is an internal error. Check the error message displayed immediately before this message. If no error message was displayed, contact the customer support center.

KFRB00159-E

Definition server process is running. (C)

The import definition server process is running.

S: Ignores the request.

KFRB00160-E

Definition server process has been suspended. (C)

The import definition server process is inactive.

S: Ignores the request.

KFRB00161-E

Failed to initialize the process, function name = *aa...aa*, errno = *xx...xx* (L)

Process initialization failed.

aa...aa: Function name

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

For details about the error number, see `errno.h` or the OS documentation.

KFRB00162-W

Synchronous reflection group is running, group name = *aa...aa*. (C)

The synchronous import group is running.

aa...aa: Synchronous import group name

S: Resumes processing.

O: Use the `hdsstate` command to check the operating status of the synchronization managing process (`hdsrefsync`) and execute the `hdsrfctl` command only when the synchronization managing process is inactive.

KFRB00163-W

Synchronous reflection group is not running, group name = *aa...aa*. (C)

The synchronous import group is not running.

aa...aa: Synchronous import group name

S: Cancels processing.

O: Use the `hdsstate` command to check the operating status of the synchronization managing process (`hdsrefsync`) and execute the `hdsrfctl` command only when the synchronization managing process is in `stop` or `not active` status.

KFRB00164-E

Synchronous reflection group doesn't exist or is canceled, group name = *aa...aa*. (C)

The `hdsrfctl` command cannot be accepted for one of the following reasons:

- The `syncgroup001` operand is missing.
- The specified synchronous import group name is not specified in the `syncgroup001` operand.
- The specified synchronous import group has been cancelled by the `hdsstart -c` command.

aa...aa: Name of the synchronous import group specified in `hdsrfctl`

S: Resumes processing. However, the `hdsrfctl` command is cancelled.

O: If you specified a synchronous import group name that was not specified in the `syncgroup001` operand, check and, if necessary, revise the synchronous import group name in the `hdsrfctl` command, and then re-execute the command.

KFRB00400-E

Failed to connect service control manager, code = *aa...aa*, errno = *xx...xx*. (E)

Datareplicator failed to establish connection with the service control manager.

aa...aa: Code

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

If you install the Oracle extraction facility or SQL Server extraction facility, both of which are HiRDB Datareplicator Extensions, the extraction service of the existing HiRDB Datareplicator is disabled. If you execute the `hdestart` command in that status, the command outputs `code=3` and `errno=1058`.

If this error occurred for the above reason, take the following action:

- To start the Oracle extraction facility or SQL Server extraction facility
If you are using the Oracle extraction facility, execute the `hdestartO` command; if you are using the SQL Server extraction facility, execute the `hdestartS` command; do not use the `hdestart` command.
- To start the HiRDB extraction facility
Reset the startup type of **HiRDB Datareplicator (Source site)** to **Manual**, or uninstall HiRDB Datareplicator Extension.

KFRB00401-E

Failed to register service handler function, `errno = xx...xx`. (E)

Registration of a service control handler failed.

`xx...xx`: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00402-E

Failed to create event object, function name = `aa...aa`, event name = `bb...bb`, `errno = xx...xx`. (E)

Event object creation processing failed.

`aa...aa`: Function name

`bb...bb`: Event name

`xx...xx`: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00403-E

Failed to open event object, function name = `aa...aa`, event name = `bb...bb`, `errno = xx...xx`. (E)

An open error occurred with an event.

aa...aa: Function name

bb...bb: Event name

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00404-E

Failed to set state of event object to signaled, function name = *aa...aa*, event name = *bb...bb*, `errno` = *xx...xx*. (E)

Event setup failed.

aa...aa: Function name

bb...bb: Event name

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00405-E

Failed to create thread to watch master, `errno` = *xx...xx*. (E)

Creation of a master monitoring thread failed.

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00406-E

Failed to create thread to watch receive master, `errno` = *xx...xx*. (E)

Creation of a reception master monitoring thread failed.

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00407-E

Failed to create thread to watch definition server, `errno` = *xx...xx*. (E)

Creation of a definition server monitoring thread failed.

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00408-E

Failed to watch master, `errno = xx...xx`. (E)

Monitoring of the master failed.

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00409-E

Failed to watch receive master, `errno = xx...xx`. (E)

Monitoring of the reception master failed.

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00410-E

Failed to watch definition server, `errno = xx...xx`. (E)

Monitoring of the definition server failed.

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00411-E

Failed to duplicate handle, `errno = xx...xx`. (E)

Duplication of a handle failed.

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00412-E

Failed to create named pipe, pipe name = *aa...aa*, errno = *xx...xx*. (E)

Creation of the named pipe failed.

aa...aa: Name of the pipe

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00413-E

Failed to connect named pipe, pipe name = *aa...aa*, errno = *xx...xx*. (E)

A connection error occurred with the named pipe.

aa...aa: Name of the pipe

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00414-E

Failed to open named pipe, pipe name = *aa...aa*, errno = *xx...xx*. (E)

An open error occurred with the named pipe.

aa...aa: Name of the pipe

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00415-E

Target Site HiRDB Datareplicator service start failed. (S)

Startup of the HiRDB Datareplicator import service failed.

Import service start processing failed.

S: Cancels processing.

O: Eliminate the cause of the error and restart the import service.

KFRB00416-E

Invalid setup information. (E)

Setup information is invalid.

S: Cancels processing.

O: Reinstall Datareplicator.

KFRB00417-E

Failed to wait for event object, function name = *aa...aa*, event name = *bb...bb*, errno = *xx...xx*. (E)

Event wait processing failed.

aa...aa: Function name

bb...bb: Event name

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Reinstall Datareplicator.

KFRB00418-E

Failed to retrieve the process termination status, process name = *aa...aa*, errno = *xx...xx*. (E)

Acquisition of a process's termination status failed.

aa...aa: Process name

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00419-E

Failed to get master process handle, errno = *xx...xx*. (E)

Acquisition of the extraction master process handle or import master process handle failed.

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00420-E

Source Site HiRDB Datareplicator service start failed. (S)

Extraction service start processing failed.

S: Cancels processing.

O: Eliminate the cause of the error and restart the extraction service.

KFRB00421-E

Failed to load library, library name = *aa...aa*, errno = *xx...xx*. (C)

Library load processing by LoadLibrary() failed.

aa...aa: Library name

xx...xx: Error number obtained by GetLastError()

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00422-E

Failed to get procedure address, procedure name = *aa...aa*, errno = *xx...xx*. (S)

Acquisition of a function address by GetProcAddress() failed.

aa...aa: Function name

xx...xx: Error number obtained by GetLastError()

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00423-E

Failed to generate Target Site HiRDB Datareplicator service stop process, errno = *xx...xx*. (S)

Generation of a target Datareplicator termination process resulted in an error.

xx...xx: Error number obtained by GetLastError()

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00424-E

Failed to generate Source Site HiRDB Datareplicator service stop process, errno = *xx...xx*. (S)

Generation of a source Datareplicator termination processing resulted in an error.

xx...xx: Error number obtained by GetLastError()

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00425-E

Failed to create child process, process name=*aa...aa*, errno = *xx...xx*.
(C)

Generation of a subprocess failed.

aa...aa: Process name

xx...xx: Error number obtained by `GetLastError()`

S: Resumes processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00426-E

Failed to create trace receive thread, errno = *xx...xx*. (C)

Creation of an activity trace reception thread failed.

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00427-E

Failed to create timer watch thread, errno = *xx...xx*. (C)

Creation of a timer monitoring thread failed.

xx...xx: Error number obtained by `GetLastError()`

S: Resumes processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00428-E

Failed to watch trace receive thread, errno = *xx...xx*. (C)

Monitoring of the activity trace reception thread failed.

xx...xx: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00429-W

Activity trace collection process check failure, function name = *aa...aa*, errno = *xx...xx*. (C)

Activity trace information will not be collected after this message has been output

because a check for the existence of the activity trace collection process failed.

aa...aa: Name of the function resulting in the error

xx...xx: Error number obtained by `GetLastError()`

S: Resumes processing.

O: If you want to collect activity trace information, terminate Datareplicator, eliminate the cause of the error on the basis of the name of the function resulting in the error and the error number obtained by `GetLastError()` by referencing the OS documentation. After you have eliminated the cause of the error, restart Datareplicator.

KFRB00430-W

Activity trace collection process does not exist. (C)

Activity trace information will not be collected after this message has been output because the activity trace collection process does not exist.

S: Resumes processing.

O: If an error message was issued by the activity trace collection process before this message, correct that error. If no such error message was issued from the activity trace collection process, contact the customer support center.

KFRB00499-E

Fatal error occurred. *xx...xx*. (C)

A fatal error occurred.

xx...xx: Error information

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00501-I

Command request was accepted. request = *aa...a* (C)

A command request was accepted.

aa...a: Request type:

INIT: Initialization request

START: Start request

NODE_START: Start request for a node

STOP: Termination request

NODE_STOP: Termination request for a node

STATE: Status acquisition request

CHG_STATUS: Status update request

S: Resumes processing.

KFRB00502-I

Source site Datareplicator start completed. (C)

Source Datareplicator startup processing has been completed.

S: Resumes processing.

KFRB00503-I

Source site Datareplicator termination completed. (C)

Source Datareplicator termination processing has been completed.

S: Terminates the source Datareplicator

KFRB00504-I

Source site Datareplicator initialization completed. (C)

Source Datareplicator initialization processing has been completed.

S: Terminates the source Datareplicator

KFRB00505-I

Command request completed. (C)

Command request processing has been completed.

S: Resumes processing.

KFRB00506-W

Source site Datareplicator start is incomplete. (C)

Source Datareplicator startup processing cannot be completed.

S: Resumes processing.

O: Check the error information for an inactive unit, eliminate the cause of the error, and then restart the source Datareplicator.

KFRB00507-W

Source site Datareplicator termination is incomplete. (C)

Source Datareplicator termination processing cannot be completed.

S: Resumes processing.

O: Check the error information for the unit that did not terminate normally.

KFRB00508-W

Source site Datareplicator initialization is incomplete. (C)

Source Datareplicator initialization processing cannot be completed.

S: Terminates the source Datareplicator.

O: Check the error information for the unit whose initialization is incomplete, eliminate the cause of the error, and then re-execute the `hdestart -i` command.

KFRB00509-W

Command request is incomplete. (C)

Command request processing cannot be completed.

S: Resumes processing.

O: Check the error information for the unit where the command request is incomplete, eliminate the cause of the error, and then re-execute the command request.

KFRB00510-I

Terminate source site Datareplicator. (C)

The source Datareplicator is being terminated.

S: Terminates the source Datareplicator.

KFRB00511-I

Preprocess file format conversion started, old format version = *aa...aa*, new format version = *bb...bb*. (C)

Format conversion of the preprocessing file has started.

aa...aa: Old format version

bb...bb: New format version

S: Resumes processing.

KFRB00512-W

-S option value of `hdestart` command is invalid, value = *aa...aa*. (C)

The value specified with the -S option in the `hdestart` command is invalid.

aa...aa: Target identifier number specified with the -S option

S: Cancels processing.

O: Because the specified value exceeds the current number of defined destinations, change the value and re-execute the command.

KFRB00513-W

-S option value of `hdestart` command is not equal to send number of incomplete initialization, previous value = *aa...aa*, current value = *bb...bb*. (C)

The value specified in the `hdestart` command's -S option does not match the target identifier number specified for the partial initialization that was not completed.

aa...aa: Target identifier number specified previously

bb...bb: Target identifier number specified currently

S: Cancels processing.

O: Specify the previously specified target identifier number and re-execute partial initialization; or, initialize the entire source Datareplicator.

KFRB00514-I

Connection request from Node-master process was accepted. (C)

A connection request from the extraction node master process was accepted.

S: Resumes processing.

KFRB00515-W

Unable to get connection information from Node-master process.
(C)

Acquisition of connection information from the extraction node master process failed.

When this message is displayed, no connection host name has been acquired from the extraction node master process, so "UNKNOWN HOST" is displayed as the host name in the error message that is sent or received before or after this message.

S: Resumes processing.

O: Take appropriate action on the basis of the received error message that was displayed immediately before this message.

KFRB00516-W

Invalid connection information from Node-master process,
hostname = *aa...aa*, reason = *bb...bb*. (C)

A connection request from the extraction node master process is invalid.

aa...aa: Name of host to be connected

bb...bb: Reason code:

1. The control unit for the extraction node master process specified in the `nodecontrol` operand does not match the control unit requested for connection establishment.
2. A connection request with a nonexistent target name was received from the extraction node master process.
3. The extraction master process was started by a transmission process start request, but a connection request for starting an update-SQL output process was received from the extraction node master process.
4. The extraction master process was started by an update-SQL output process start request, but a connection request for starting a transmission process was received from the extraction node master process.
5. An error was detected while Datareplicator was checking the connection status with the extraction node master process.

6. An error was detected during node initialization processing.
7. An error was detected during definition information transmission processing on the extraction node master process.

S: Resumes processing.

O: Take the following action:

Reason code	Description	Action
1	The control unit for the extraction node master process specified in the <code>nodecontrol</code> operand does not match the control unit requested for connection establishment.	If <code>nodecontrol</code> is <code>unit</code> , extraction node master processes cannot be started individually. To start an extraction node master process, execute the <code>hdestart</code> command on the extraction master process.
2	A connection request with a nonexistent target name was received from the extraction node master process.	Specify the correct server name in the <code>hdestart_n</code> command, and then re-execute the command.
3	The extraction master process was started by a transmission process start request, but a connection request for starting an update-SQL output process was received from the extraction node master process.	To re-execute update-SQL output processing, terminate the source Datareplicator, and then specify <code>-L</code> in the <code>-s</code> operand of the <code>hdestart</code> command for the extraction master process.
4	The extraction master process was started by an update-SQL output process start request, but a connection request for starting a transmission process was received from the extraction node master process.	
5	An error was detected while Datareplicator was checking the connection status with the extraction node master process.	Take appropriate action according to the error message that was displayed immediately before this message.
6	An error was detected during initialization processing for a node.	
7	An error was detected during definition information transmission processing on the extraction node master process.	Take appropriate action according to the error message that was displayed immediately before this message. If the extraction node master process has resulted in a status file open error, the <code>hdestart_n</code> command's execution unit might be invalid. Re-execute the command in the correct unit.

KFRB00517-W

Node-master process is already connected, target = *aa...aa*, current hostname = *bb...bb*, request hostname = *cc...cc*. (C)

The extraction node master process specified in the connection request has already been connected.

aa...aa: Name of the connection target

bb...bb: Name of the host currently connected

cc...cc: Name of the new host specified in the connection request

S: Resumes processing.

KFRB00518-I

Terminate Extract master process, because connection wait time from Node-master process was over. (C)

The extraction master process is now terminated because the connection wait time from the extraction node master process was reached.

S: Terminates the system.

KFRB00519-I

Connection request completed, target = *aa...aa*, connect hostname = *bb...bb*. (C)

Connection processing from the extraction node master process was completed.

aa...aa: Name of the connection target

bb...bb: Name of the host being connected

S: Resumes processing.

KFRB00551-I

Extract master process request was accepted, request = *aa...a*. (C)

A request from the extraction master process was accepted.

aa...a: Request type:

INITENV: Environment creation request

START: Start request

STOP: Termination request

STATE: Status acquisition request

CHG_STATUS: Status update request

S: Resumes processing.

KFRB00552-I

Node-master process start completed. (C)

Startup of the extraction node master process has been completed.

S: Resumes processing.

KFRB00553-I

Node-master process termination completed. (C)

Termination processing has been completed for the extraction node master process.

S: Resumes processing.

KFRB00554-I

Node-master process initialization completed. (C)

Initialization processing has been completed for the extraction node master process.

S: Terminates the corresponding node processing.

KFRB00555-I

Extract master process request completed. (C)

Extraction master process request processing has been completed.

S: Resumes processing.

KFRB00556-W

Extract master process request is incomplete. (C)

Extraction master process request processing cannot be completed.

S: Resumes processing.

O: Check the messages that have been output.

KFRB00557-I

Terminate node-master process. (C)

The extraction node master process is being terminated.

S: Terminates the corresponding node processing.

KFRB00558-W

Connection was closed, Node-master process retry connect to Extract master process. (C+L)

Connection was broken. Attempt will be made to reconnect to the extraction master process.

S: Resumes processing.

O: Check the previous message to determine the cause of the communication error. Also check the status of the extraction master process and restart it, if necessary.

KFRB00559-W

Connection wait time to Extract master process was over. (C+L)

The maximum wait time for connection to the extraction master process has elapsed.

S: Cancels processing for the affected node.

O: Check the previous message to determine the cause of the communication error.

KFRB00560-E

Connection request to Extract master process was not accepted,
message number = *nnnnn*. (C+L)

A connection request to the extraction master process was not accepted.

nnnnn: Message number for the error that occurred in the extraction master process

nnnnn corresponds to *KFRBnnnnn-T*, a message output by Datareplicator.

S: Cancels processing for the affected node.

O: See the description of the *KFRBnnnnn-T* message that corresponds to message number *nnnnn*.

KFRB00561-E

Node-master process connected to Extract master process, host
name = *aa...aa*, port number = *bb...bb*. (C+L)

Connection processing on the extraction master process was completed.

aa...aa: Name of the host to be connected

bb...bb: Port number

S: Resumes processing.

KFRB00562-I

aa...aa Function applied, code = *xx*. (C)

Function *aa...aa* is being applied.

aa...aa: One of the following functions that is being applied:

High-availability:

Improvement of the reliability of *syslogfile*

High-availability and Code conversion:

Improvement of the reliability of *syslogfile* and character encoding
conversion (from SJIS to UTF-8) on *syslogfile*

xx: Detail code

01: The functions for improving the reliability of *syslogfile* and for
performing character encoding conversion (from SJIS to UTF-8) on
syslogfile are both being used.

02: For one of the following reasons, only the function for improving the
reliability of *syslogfile* is being used, without using the function for
performing character encoding conversion on *syslogfile*:

- The Hitachi encoding conversion component has not been installed.
- Initialization of the Hitachi encoding conversion component failed. A

memory shortage might have occurred.

03: The function for performing character encoding conversion on `syslogfile` is not being used because the extended `syslog` facility does not support the encoding conversion function although the `dblocale` and `msglocale` operand values in Datareplicator's extraction system definition or import system definition are appropriate. Only the function for improving the reliability of `syslogfile` is being used.

04: The function for performing character encoding conversion on `syslogfile` is not being used because the combination of the `dblocale` and `msglocale` operand values in Datareplicator's extraction system definition or import system definition is not supported by the encoding conversion function. Only the function for improving the reliability of `syslogfile` is used.

S: Resumes processing.

[Action]

The following table shows the detail code and the action when the character encoding conversion functionality is being used on `syslogfile` on an OS that supports character encoding conversion of `syslogfile`.

Detail code	Action
01	No action is needed.
02	When the Hitachi encoding conversion component is not installed: Terminate Datareplicator, install the Hitachi encoding conversion component, and then restart Datareplicator. When the Hitachi encoding conversion component is installed: <ul style="list-style-type: none"> • If a memory shortage occurred, resolve the memory shortage, and then restart Datareplicator. • For any other problem, terminate Datareplicator, reinstall the Hitachi encoding conversion component, and then restart Datareplicator. If this action does not resolve the problem, contact the customer support center.
03	Terminate Datareplicator, install the extended <code>syslog</code> facility that supports the character encoding conversion function, and then restart Datareplicator.
04	Terminate Datareplicator, correct the combination of the <code>dblocale</code> and <code>msglocale</code> operand values in Datareplicator's extraction system definition or import system definition so that the combination is supported by the character encoding conversion function, and then start Datareplicator.

KFRB00581-I

Interrupted request of forceful termination. (C)

Processing was interrupted by a forced termination request.

S: Terminates the source Datareplicator by forced termination processing.

KFRB00582-I

Communicable node-master process not exist. (C)

There is no extraction node master process available for communication.

S: Terminates the source Datareplicator.

KFRB00583-I

Capture process and Sender process not exist. (C)

There is neither an extraction process nor a transmission process.

S: Terminates the extraction node master process.

KFRB00584-I

Status update completed, server name = *aa...a* (C)

Updating of the status of the server indicated in the message was successful.

aa...aa: Server name

S: Resumes processing.

KFRB00601-E

Create socket failed, errno = *xx...xx*. (C)

Socket creation failed.

xx...xx: Error number set in `errno`

S: Cancels the corresponding node processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00602-E

Unable to get host-information, hostname = *aa...aa*, `h_errno` = *xx...xx*. (C)

Acquisition of host information failed.

aa...aa: Host name

xx...xx: Error number set in `errno`

S: Cancels the corresponding node processing.

O: Make sure that the *aa...aa* service has been registered correctly into the OS's `services` file.

KFRB00603-E

Connection failed, hostname = *aa...aa*, errno = *xx...xx*. (C)

A connection request failed.

aa...aa: Host name

xx...xx: Error number set in `errno`

S: Cancels the corresponding node processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00604-E

Send error, hostname = *aa...aa*, `errno` = *xx...xx*. (C)

Transmission processing failed.

aa...aa: Host name

xx...xx: Error number set in `errno`

S: Cancels the corresponding node processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00605-E

Receive error, hostname = *aa...aa*, `errno` = *xx...xx*. (C)

Reception processing failed.

aa...aa: Host name

xx...xx: Error number set in `errno`

S: Cancels the corresponding node processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00606-E

Send timeout, hostname = *aa...aa*. (C)

Datareplicator was unable to transmit data within the communication wait time.

aa...aa: Host name

S: Cancels the corresponding node processing.

O: Check that the extraction node master process is active, or check for a communication error between the hosts.

KFRB00607-E

Receive timeout, hostname = *aa...aa*. (C)

Datareplicator was not able to receive data within the communication wait time.

aa...aa: Host name

S: Cancels the corresponding node processing.

O: Check that the extraction node master process is active, or check for a

communication error between the hosts.

KFRB00608-E

Found end of transmission, hostname = *aa...aa*. (C)

End of communication was detected.

aa...aa: Host name

S: Cancels the corresponding node processing.

O: Check that the extraction node master process is active, or check for a communication error between the hosts.

KFRB00609-E

Unable to get service-information, service name = *aa...aa*, errno = *xx...xx*. (E)

Datareplicator was not able to obtain service information.

aa...aa: Service name

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00610-E

Bind system call error, errno = *xx...xx*. (E)

The bind system call resulted in an error.

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00611-E

Listen system call error, errno = *xx...xx*. (C)

The listen system call resulted in an error.

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00612-E

Accept system call error, errno = *xx...xx*. (C)

The accept system call resulted in an error.

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00613-E

Setsockopt system call error, `errno = xx...xx`. (C)

An error occurred in the `setsockopt` system call.

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

For details about the error number, see `errno.h` or the OS documentation.

KFRB00614-E

Select system call error, `errno = xx...xx`. (C)

An error occurred in the `select` system call.

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

For details about the error number, see `errno.h` or the OS documentation.

KFRB00700-W

Warning occurred on node-master process, `hostname = aa...aa`, server name = `bb...bb`, `senderid = cc...cc`, message number = `nnnnn`. (C+E)

A warning occurred in the extraction node master process.

aa...aa: Host name

bb...bb: Server name

cc...cc: Identifier of the destination

nnnnn: Message number of the warning that occurred in the extraction node master process

nnnnn corresponds to `KFRBnnnnn-T`, a message output by Datareplicator.

S: See the description of the `KFRBnnnnn-T` message that corresponds to message number *nnnnn*.

O: See the description of the `KFRBnnnnn-T` message that corresponds to message number *nnnnn*.

KFRB00701-E

Error occurred on node-master process, hostname = *aa...aa*, server name = *bb...bb*, senderid = *cc...cc*, message number = *nnnnn*. (C)

An error occurred in the extraction node master process.

aa...aa: Host name

bb...bb: Server name

cc...cc: Target identifier

nnnnn: Message number for the error resulting from the extraction node master process

nnnnn corresponds to KFRB*nnnnn*-T, a message output by Datareplicator.

S: See the description of the KFRB*nnnnn*-T message that corresponds to message number *nnnnn*.

O: See the description of the KFRB*nnnnn*-T message that corresponds to message number *nnnnn*.

KFRB00702-E

Missing necessary file, file name = *aa...aa*, file-kind = *bb...bb*. (C)

A required file is missing.

aa...aa: Filename

bb...bb: File type:

EXTDEF (extraction definition preprocessing file)

MSTSTATFILE (extraction master status file)

NMTSTATFILE (extraction server status file)

ERRFILE (error information file at the source)

ERRTXT (message text file)

S: Cancels processing.

O: Provide the file and re-execute. The message text file is created during installation; if it is missing, you must reinstall Datareplicator.

KFRB00703-E

File initialization error, file name = *aa...aa*, file-kind = *bb...bb*, errno = *xx...xx*. (C)

File initialization processing resulted in an error.

aa...aa: Filename

bb...bb: File type:

MSTSTATFILE (extraction master status file)

NMTSTATFILE (extraction server status file)

QUEFILE (extraction information queue file)

CNCTFILE (data linkage file)

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00704-E

Invalid process communication, hostname = *aa...aa*. (C)

Communication was established with an invalid process. host name = *aa...aa*

aa...aa: Host name

S: Cancels the corresponding node processing.

O: Check the network settings for errors.

KFRB00705-E

Node-master process is running. (C)

The extraction node master process is running.

S: Ignores the request.

KFRB00706-E

Invalid request was accepted, hostname = *aa...aa*. (C)

An invalid request was accepted.

aa...aa: Host name

S: Cancels the corresponding node processing.

O: This is an internal error. Contact the customer support center.

KFRB00708-E

Process terminated due to inner error, process = *aa...aa*. (C)

A process terminated abnormally.

aa...aa: Process name

S: Cancels the corresponding node processing.

O: This is an internal error. Contact the customer support center.

KFRB00709-E

Missing necessary environment variable value, variable name = *aa...aa*. (C)

Datareplicator was unable to obtain the value of an environment variable.

aa...aa: Environment variable name

S: Cancels processing.

O: Specify the environment variable.

KFRB00710-E

Invalid environment variable value, variable name = *aa...aa*. (C)

The value of an environment variable is invalid.

aa...aa: Environment variable name

S: Cancels processing.

O: Check the environment variable and correct its value.

KFRB00711-E

Invalid server name specification, server name = *aa...aa*. (C)

An invalid server name is specified in the command.

aa...aa: Server name

S: Cancels the processing only for the invalid server name.

O: Specify the correct argument and re-execute.

KFRB00712-E

Invalid sender-id specification, sender-id = *aa...aa*. (C)

An invalid target identifier is specified in the command.

aa...aa: Target identifier

S: Cancels the processing only for the invalid target identifier.

O: Specify the correct argument and re-execute.

KFRB00713-E

Unable to load preprocess file of extract definition, because preprocess file is older than extract master status file. (C)

Datareplicator cannot read the extraction definition preprocessing file because it is older than the extraction master status file.

S: Cancels processing.

O: Execute the `hdeprep` command to re-create the extraction definition preprocessing file.

KFRB00714-E

Unable to use file system, file system = *aa...aa*, errcode = *bb...bb*. (L)

Datareplicator was unable to use a file system area.

aa...aa: Name of the Datareplicator file system area

bb...bb: Detail code for the Datareplicator file system area access error

S: Cancels processing.

O: Check that the file specified for the Datareplicator file system area has been initialized, and then re-execute. The following table explains the detail codes for a Datareplicator file system area access error:

Detail code	Description
1	Specified file is not a character special file.
2	The Datareplicator file system area has not been initialized.
3	The Datareplicator file system area has already been initialized.
4	The Datareplicator file system area is in use (inactive).
5	The Datareplicator file system area is in use (active).
6	Attempt was made to register more than one file with the same name in the Datareplicator file system area.
7	Attempt was made to store more than the maximum number of files permitted for the Datareplicator file system area.
8	Attempt was made to store files that exceed the actual size of the Datareplicator file system area. Re-estimate the size of the Datareplicator file system area by referring to the structure of the area, as shown in 3.5.3 <i>Structure of a Datareplicator file system area</i> .
9	There are not enough sectors in the character special file. Re-estimate the size of the Datareplicator file system area by referring to the structure of the area, as shown in 3.5.3 <i>Structure of a Datareplicator file system area</i> .
10	A system call error occurred. Possible reasons are: <ul style="list-style-type: none"> • The Datareplicator file system area was not found. • The Datareplicator file system area has been damaged.
11	Link file creation failed. Possible reasons are: <ul style="list-style-type: none"> • The file to be stored in the Datareplicator file system area already exists. • The path on which the link file is to be created does not exist or there is no access permission.
12	Memory shortage occurred.
13	The sector length of the character special file is too small (it must be at least 256 bytes).

KFRB00715-E

Unable to initialize file system, file system = *aa...aa*, errcode = *bb...bb*. (L)

Initialization of a file system area failed.

aa...aa: Name of the Datareplicator file system area

bb...bb: Detail code for the Datareplicator file system area access error

S: Cancels processing.

O: Initialize the file specified for the Datareplicator file system area using the `hdsfmkfs` command, and then re-execute. For details about the detail codes for a Datareplicator file system area access error, see *KFRB00714-E*.

KFRB00716-E

Unable to prepare file system, file system = *aa...aa*, errcode = *bb...bb*. (L)

Preparation of a file system area failed.

aa...aa: Name of the Datareplicator file system area

bb...bb: Detail code for the Datareplicator file system area access error

S: Cancels that the file specified for the Datareplicator file system area is usable, and then re-execute. For details about the detail codes for a Datareplicator file system area access error, see *KFRB00714-E*.

KFRB00717-E

Unable to add file system, file system = *aa...aa*, add file = *bb...bb*, add size = *cc...cc*, errcode = *dd...dd*. (L)

Datareplicator was unable to add a file to a file system area.

aa...aa: Name of the Datareplicator file system area

bb...bb: Name of the file to be added to the Datareplicator file system area

cc...cc: Size of the file to be added

dd...dd: Detail code for the Datareplicator file system area access error

S: Cancels processing.

O: Check the size of the file to be stored in the Datareplicator file system area, and then re-execute. For details about the detail codes for a Datareplicator file system area access error, see *KFRB00714-E*.

KFRB00718-E

Unable to use both UNIX regular file and Character special file. (S)

Datareplicator cannot use both UNIX files and character special files.

S: Cancels processing.

O: Correct the type of file to be used, and then re-execute.

KFRB00719-E

Source site database is not supported HiRDB, HiRDB = *aa...aa*,
 Datareplicator = *bb...bb*. (L)

The source database is not a supported HiRDB.

aa...aa: Source HiRDB's addressing mode

bb...bb: Source Datareplicator's addressing mode

S: Cancels processing.

O: Check that Datareplicator that has been installed is supported by the source HiRDB.

KFRB00720-E

Failed to get information of source site database, function name
 = *aa...aa*, errno = *bb...bb*. (L)

Acquisition of source database information failed.

aa...aa: Function name

bb...bb: Error number

S: Cancels processing.

O: This is an internal error. Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00721-E

Unable to get current host name, target name = *aa...aa*, information
 = *bb...bb*. (C)

Acquisition of the current host name failed.

aa...aa: Name of the target

bb...bb: Error information

S: Cancels processing.

O: Eliminate the cause of the error on the basis of the error information, and then re-execute.

If the error information displays "Unable to get host name.
 rtncode=-1854", start the source HiRDB, and then re-execute the command.

KFRB00722-E

Command request is invalid, reason = *aa...aa*. (C)

The command request is invalid.

aa...aa: Reason code:

Unmatch nodecontrol operand: The nodecontrol operand does not match the command.

S: Cancels processing.

O: If the `nodecontrol` operand value is `unit`, the `hdestart_n` command cannot be used to start the extraction master process. In such a case, use the `hdestart` command to start the extraction master process.

KFRB00723-E

Unable to access shared memory, because the exclusion could not get, type = *aa...aa*. (C+L)

Access to the shared memory failed because the system was unable to acquire a lock.

aa...aa: Type of shared memory:

COMMON (shared memory common to import processing)

DEFSERV (shared memory for import definition analysis)

EXTCOM (shared memory common to extraction processing)

STATINF (shared memory for status information)

EXTDEF (shared memory for extraction definition analysis)

ERRTXT (shared memory for message text)

LOBSHM (shared memory for importing BLOB)

NMTCMD (shared memory for communication between extraction node master and command)

S: Cancels processing.

O: If a command is executing in the source system, restart the extraction node master process after that command has terminated.

KFRB00724-W

Status update is incompleted, host name = *aa...aa*. (C)

Status updating failed for the node that corresponds to the host whose name is displayed in the message.

aa...aa: Host name

S: Resumes processing.

O: Take appropriate action according to the error message that was displayed immediately before this message. If no error message has been displayed, take appropriate action according to the `KFRB00725-W` message that has been output to the corresponding node's extraction node master error information file.

KFRB00725-W

Status update is incompleted, server name = *aa...aa*. (C)

Status updating failed for the server whose name is displayed in the message.

aa...aa: Server name

S: Resumes processing.

O: Take appropriate action according to the error message that was displayed immediately before this message.

KFRB00726-E

Invalid operand *pd_rpl_reflect_mode* of source *HiRDB*. (C)

To use the import transaction synchronization facility, you must specify *pd_rpl_reflect_mode=uap* in the source *HiRDB*'s system definition.

S: Cancels processing.

O: Specify *pd_rpl_reflect_mode=uap* in the *HiRDB* system definition, and then restart *HiRDB*.

KFRB00801-E

Invalid *HiRDB* Datareplicator-id, id = *aa*, definition file = *bb...bb*. (C)

Specified source Datareplicator identifier is invalid.

aa: Source Datareplicator identifier

bb...bb: Name of the extraction system definition file

S: Cancels processing.

O: Check the variable in the extraction system definition file and correct its value.

KFRB00802-E

Invalid extract environment statement file name in definition file, file name = *aa...aa*, definition file = *bb...bb*. (C)

The name of the extraction environment definition file in the extraction system definition file is invalid.

aa...aa: Name of the extraction environment definition file

bb...bb: Name of the extraction system definition file

S: Cancels processing.

O: Check the variable in the extraction system definition file and correct its value.

KFRB00803-E

Invalid replication sender-id in definition file, sender-id = *aa...aa*, definition file = *bb...bb*. (C)

A target identifier is invalid.

aa...aa: Target identifier

bb...bb: Name of the extraction system definition file

S: Cancels processing.

O: Check the variable in the extraction system definition file and correct its value.

KFRB00804-E

Duplicate replication sender-id in definition file, sender-id = *aa...aa*, definition file = *bb...bb*. (C)

A target identifier is duplicated.

aa...aa: Target identifier

bb...bb: Name of the extraction system definition file

S: Cancels processing.

O: Check the variable in the extraction system definition file and correct its value.

KFRB00805-E

Missing transmission environment statement file in definition file, definition file = *aa...aa*. (C)

There are no transmission environment definition files.

aa...aa: Name of the extraction system definition file

S: Cancels processing.

O: Check the variables in the extraction system definition file and correct their values.

KFRB00806-E

Invalid transmission environment statement file name in definition file, file name = *aa...aa*, definition file = *bb...bb*. (C)

The name of a transmission environment definition file is invalid.

aa...aa: Name of the transmission environment definition file

bb...bb: Name of the extraction system definition file

S: Cancels processing.

O: Check the variable in the extraction system definition file and correct its value.

KFRB00807-E

Missing replication node-id in definition file, definition file = *aa...aa*. (C)

No data linkage identifier was specified.

aa...aa: Name of the extraction environment definition file

S: Cancels processing.

O: Check the variable in the extraction environment definition file and correct its value.

KFRB00808-E

Service name is not found in services file, service name = *aa...aa*,
definition file = *bb...bb*. (C)

The specified service name is not registered.

aa...aa: Service name

bb...bb: Name of the extraction system definition file or transmission environment
definition file

S: Cancels processing.

O: Register the service name or specify a service name that is registered.

KFRB00809-E

Missing extract environment statement file in definition file,
definition file = *aa...aa*. (C)

An extraction environment definition file is missing.

aa...aa: Name of the extraction system definition file

S: Cancels processing.

O: Check the variable in the extraction system definition file and correct its value.

KFRB00813-E

Duplicate transmission queue file name in definition file, file
name = *aa...aa*, definition file = *bb...bb*. (C)

The name of an extraction information queue file is duplicated.

aa...aa: Name of the extraction information queue file

bb...bb: Name of the extraction environment definition file

S: Cancels processing.

O: Check the variable in the extraction environment definition file and correct its
value.

KFRB00814-E

Invalid file name in definition file, file name = *aa...aa*,
definition file = *bb...bb*. (C)

A filename is invalid.

aa...aa: Invalid filename

bb...bb: Definition filename

S: Cancels processing.

O: Check the variable in the definition file and correct its value.

KFRB00815-E

Transmission queue file must be more than two in definition file,
definition file = *aa...aa*. (C)

No more than one extraction information queue file was specified.

aa...aa: Name of the extraction environment definition file

S: Cancels processing.

O: Specify at least two extraction information file queue files in the extraction environment definition file.

KFRB00818-E

Invalid service name in definition file, service name = *aa...aa*,
definition file = *bb...bb*. (C)

A service name is invalid.

aa...aa: Service name

bb...bb: Name of the extraction system definition file or transmission environment definition file

S: Cancels processing.

O: Check the variable in the extraction system definition file or transmission environment definition file and correct its value.

KFRB00819-E

Invalid host name in definition file, host name = *aa...aa*,
definition file = *bb...bb*. (C)

A host name is invalid.

aa...aa: Host name

bb...bb: Name of the extraction system definition file or transmission environment definition file

S: Cancels processing.

O: Check the host name specified in the definition file and correct it.

KFRB00820-E

Invalid nsnd-id, nsnd-id = *aa*, definition file = *bb...bb*. (C)

A specified transmission-suppressed original receiver identifier is invalid.

aa: Transmission-suppressed original receiver identifier

bb...bb: Name of the transmission environment definition file

S: Cancels processing.

O: Check the variable in the transmission environment definition file and correct its

value.

KFRB00821-E

Duplicate nsnd-id in definition file, definition file = *aa...aa*. (C)

A transmission-suppressed original receiver identifier is duplicated.

aa...aa: Name of the transmission environment definition file

S: Cancels processing.

O: Check the variable in the transmission environment definition file and correct its value.

KFRB00822-E

Missing HiRDB Datareplicator-id, definition file = *aa...aa*. (C)

No source Datareplicator identifier was specified.

aa...aa: Name of the extraction system definition file

S: Cancels processing.

O: Check the variable in the extraction system definition file and correct its value.

KFRB00823-E

Queue file I/O buffer size in definition file exceeds transmission queue file size, definition file = *aa...aa*. (C)

The extraction information queue file I/O buffer is larger than the extraction information queue file.

aa...aa: Name of the extraction environment definition file

S: Cancels processing.

O: Check the variable in the extraction environment definition file and correct its value.

KFRB00824-E

Variable was changed in definition file after last information, definition file = *aa...aa*, variable = *bb...bb*. (C)

The contents of a definition file were changed during restart.

aa...aa: Definition filename

bb...bb: Variable name

S: Cancels processing.

O: Execute the `hdestart -i` command to initialize the definition file or restore the previous contents of the definition file, and then restart. If you are performing partial initialization for two or more target identifiers or data linkage identifiers, do them one at a time (change one target identifier or data linkage identifier definition and perform

partial initialization for it, and then repeat for the next target identifier or data linkage identifier).

KFRB00831-E

Server Common definition get error, definition file = *aa...aa*,
errno = *xx...xx*. (C)

Acquisition of the common definition section failed.

aa...aa: Definition filename

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00832-E

Server local definition get error, definition file = *aa...aa*, errno
= *xx...xx*. (C)

Acquisition of an individual definition section failed.

aa...aa: Definition filename

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB00833-E

Missing sender-id in definition file, definition file = *aa...aa*.
(C)

No target identifier was specified.

aa...aa: Name of the extraction system definition file

S: Cancels processing.

O: Check the variable in the extraction system definition file and correct its value.

KFRB00834-E

Missing sender system-id in definition file, definition file =
aa...aa. (C)

No target system identifier was specified.

aa...aa: Name of the transmission environment definition file

S: Cancels processing.

O: Check the variable in the transmission environment definition file and correct its

value.

KFRB00835-E

Invalid header in definition file, definition file = *aa...aa*. (C)

The specified header is invalid.

aa...aa: Definition filename

S: Cancels processing.

O: Check the header in the definition file and correct the error.

KFRB00837-E

Product of maxtran and maxtrandata exceeds maximum value,
definition file = *aa...aa*. (C)

The product of the maximum number of concurrently executable transactions times the maximum number of update information items per transaction exceeds the maximum value.

aa...aa: Name of the transmission environment definition file

S: Cancels processing.

O: Check the variable in the transmission environment definition file and correct its value.

KFRB00838-E

Invalid server name in definition file, server name = *aa...aa*,
definition file = *bb...bb*. (C)

An invalid server name was specified in the header for an individual definition.

aa...aa: Server name

bb...bb: Name of the extraction environment definition file or transmission environment definition file

S: Cancels processing.

O: Check the server name in the individual definition section and correct the error.

KFRB00841-E

Invalid Tselector in definition file, Tselector = *aa...aa*,
definition file = *bb...bb*. (S)

T-selector is invalid.

aa...aa: T-selector

bb...bb: Definition filename

S: Stops processing.

O: Eliminate the cause of the error and re-execute.

KFRB00842-E

Invalid NSAP address in definition file, NSAP address = *aa...aa*,
definition file = *bb...bb*. (S)

NSAP address is invalid.

aa...aa: NSAP address

bb...bb: Definition filename

S: Stops processing.

O: Eliminate the cause of the error and re-execute.

KFRB00843-E

Cannot specify OSI protocol, definition file = *aa...aa*. (S)

OSI cannot be specified for the communications protocol.

aa...aa: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB00844-E

Invalid operand value, definition file = *aa...aa*, operand = *bb...bb*,
value = *cc...cc*. (S)

An operand value is invalid.

aa...aa: Definition filename

bb...bb: Operand name

cc...cc: Operand value

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB00845-E

Not specified Necessary operand, definition file = *aa...aa*, operand
= *bb...bb*. (S)

A required operand is missing.

aa...aa: Definition filename

bb...bb: Operand name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB00846-E

Duplicate port number, port number = *aa...aa*, definition file = *bb...bb*. (S)

A port number is duplicated.

aa...aa: Port number

bb...bb: Definition filename

S: Cancels processing.

O: The port number for the service name specified in the *mstservice* and *connection_accept_service* operands in the extraction system definition is duplicated. Correct the definition so that there is no duplication of port numbers, and then re-execute the command.

KFRB00847-E

Combination of operand *aa...a* and *bb...b* is illegal. (C+S)

A combination of operands that is not supported by the import transaction synchronization facility was detected.

aa...a: Operand name

bb...b: Operand name

S: Cancels processing.

O: Correct the operand settings, and then re-execute initialization.

KFRB00848-E

Event code not unique in definition file, definition file = *aa...aa*, operand = *bb...bb*, *cc...cc*. (L+E)

An event code is duplicated.

aa...aa: Definition filename

bb...bb: Operand name

cc...cc: Operand name

S: Cancels processing.

O: The event codes specified in the *bb...bb* and *cc...cc* operands are duplicated in definition file *aa...aa*.

Correct the definition so that no event code is duplicated, and then re-execute the command.

KFRB00849-E

Number of the servers which have extraction tables exceeds 128. (C)

When the import transaction synchronization facility is used, the number of servers

containing the target tables exceeds the maximum value.

S: Cancels processing.

O: When the import transaction synchronization facility is used, the total number of back-end servers that contain all target tables is the maximum value that can be received by a single target Datareplicator (128 when the target system is UNIX, 63 when it is Windows). Correct the extraction definition so that the number of back-end servers does not exceed the maximum value, execute the `hdeprep` command, and then re-execute the `hdestart` command.

KFRB00851-E

Missing command-option in definition file, definition file = *aa...aa*, command = *bb...bb*, option = *cc...cc*. (C)

A required command option is missing in the HiRDB's system common definition.

aa...aa: Definition filename

bb...bb: Command name

cc...cc: Option name

S: Cancels processing.

O: Check the contents of the HiRDB definition file and correct the error.

KFRB00861-E

Duplicate file kind appointed in definition file, server name = *aa...aa*, operand = *bb...bb*, definition file = *cc...cc*. (C)

A file type to be allocated to the file system area has already been allocated.

aa...aa: Server name

bb...bb: Operand name

cc...cc: Definition filename

S: Cancels processing.

O: Check the contents of the extraction environment definition file and correct the error.

KFRB00862-E

Invalid file kind appointed in definition file, file kind = *aa...aa*, definition file = *bb...bb*. (C)

A file type to be allocated to the file system area is invalid.

aa...aa: Type of file to be allocated

bb...bb: Definition filename

S: Cancels processing.

O: Check the contents of the extraction environment definition file and correct the error.

KFRB00863-E

Duplicate file system appointed in definition file, file system = *aa...aa*, definition file = *bb...bb*. (C)

A file system area name is duplicated.

aa...aa: Name of the Datareplicator file system area

bb...bb: Definition filename

S: Cancels processing.

O: Check the contents of the extraction environment definition file and correct the error.

KFRB00864-W

System-call(*aa...aa*) error, param = *bb...bb*, detail = *cc...cc*. (L)

A system call error occurred.

aa...aa: System call name

bb...bb: Parameter information[#]

cc...cc: System call error character string

S: Cancels processing.

O: Eliminate the cause of the error, and then re-execute the command.

#

For details about the relationship between the parameter information and the system call errors, see *10.3 List of system call errors*.

KFRB00865-E

System-call(*aa...aa*) error, param = *bb...bb*, errno = *cc...cc*. (L)

An error occurred in a system call (*aa...aa*).

aa...aa: System call name

bb...bb: Parameter information[#]

cc...cc: Error number

S: Cancels processing.

O: Eliminate the cause of the error on the basis of the error number, and then re-execute. For details about the error number, see `error.h` or the applicable OS documentation.

#

For details about the relationship between the parameter information and the system call errors, see *10.3 List of system call errors*.

KFRB00866-E

Changed table definition after executed `hdeprep`, table name = *aa...aa*, table-id = *bb...bb*. (C)

The table definition of the source table was changed after the `hdeprep` command was executed.

aa...aa: Table name

bb...bb: Table ID

S: Cancels processing.

O: If you have not changed the structure of the displayed table, execute the `hdeprep` command, and then re-execute the `hdestart` command. If you have changed the table structure, achieve conformity between the source and target databases, and then execute initial start on the source and target Datareplicators.

KFRB00868-W

Not check effectivity of `hde_prpfile`, reason = *aa...aa*, *bb...bb*. (C)
`hde_prpfile` is not being checked for validity.

aa...aa: Reason code 1

In the case of `HiRDB CONNECT`, startup is performed without checking the change to the table definition because `HiRDB` cannot be connected.

bb...bb: Reason code 2 (displays the `SQLCODE` when `CONNECT` was issued)

S: Resumes processing.

KFRB00869-E

Unable to allocate several server's files into same file system, server name = *aa...aa*, operand = *bb...bb*, definition file = *cc...cc*. (C)

Only a single server's files can be allocated to a file system area.

aa...aa: Server name

bb...bb: Operand name

cc...cc: Definition filename

S: Stops processing.

O: If the `nodecontrol` operand value is `server` in the extraction system definition, the same file system area cannot store files for multiple servers.

Correct the definition file so that only files for a single server are stored in the file system area, and then re-execute the command.

KFRB00870-E

Error occurred by the command execution, command = *aa...a*, code = *bb...b*. (C)

An error occurred in a command executed by Datareplicator.

aa...a: Command name

bb...b: Command's return code

S: Cancels processing.

O: Check the environment to determine whether this command can be executed. Eliminate the cause of the error and re-execute.

KFRB00915-E

Unable to open analysis file, analysis file = *aa...aa*, errno = *xx...xx*. (S+C)

An open error occurred in the analysis file that is used to analyze the definition information.

aa...aa: Name of the analysis file

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

xx...xx: Error number

If the error number is 0, the pathname is invalid.

S: Stops processing.

[Action]

Contact the system administrator to determine whether Datareplicator is installed correctly.

KFRB00916-E

Incorrect variable, file = *aa...aa*, line = *bb...bb*, variable = *cc...cc*. (S+C)

The value of a variable is invalid in the definition file.

aa...aa: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

bb...bb: Line resulting in the error

cc...cc: Variable name

S: Stops processing.

[Action]

Check the variable specified in the definition file and correct its value.

KFRB00917-E

Command name invalid, file = *aa...aa*, line = *bb...bb*, command = *cc...cc*.
(S+C)

Datareplicator is unable to analyze a command specified in the definition file.

aa...aa: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

bb...bb: Line resulting in the error

cc...cc: Command name

S: Stops processing.

[Action]

Check the command specified in the definition file and correct the error.

KFRB00918-E

Operand name in definition file invalid, file = *aa...aa*, line = *bb...bb*, operand = *cc...cc*. (S+C)

An operand name is invalid in the definition file.

aa...aa: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

bb...bb: Line resulting in the error

cc...cc: Operand name

S: Stops processing.

[Action]

Check the operand name specified in the definition file and correct the error.

KFRB00919-E

Command argument invalid, file = *aa...aa*, line = *bb...bb*, command = *cc...cc*. (S+C)

A command argument is invalid.

aa...aa: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

bb...bb: Line resulting in the error

cc...cc: Command name

S: Stops processing.

[Action]

Check the command argument specified in the definition file and correct the error.

KFRB00920-E

Operand argument invalid, file = *aa...aa*, line = *bb...bb*, operand = *cc...cc*. (S+C)

An operand argument is invalid.

aa...aa: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

bb...bb: Line resulting in the error

cc...cc: Operand name

S: Stops processing.

[Action]

Check the operand argument specified in the definition file and correct the error.

KFRB00921-E

Unable to analyze definition file due to insufficient memory, memory requirement = *aa...aa*. (S+C)

A memory shortage occurred during analysis of the definition file.

aa...aa: Required memory size at the time of the error

S: Stops processing.

[Action]

Eliminate the cause of the error and re-execute.

KFRB00922-E

Number of nests in definition file exceeds the limit, file = *aa...aa*, line = *bb...bb*. (S+C)

The number of nesting levels in the definition file exceeded the maximum value.

aa...aa: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

bb...bb: Line resulting in the error

S: Stops processing.

[Action]

Check the number of nesting levels in the definition file and correct the error.

KFRB00923-E

Operand *aa...aa* specified twice, file = *bb...bb*, line = *cc...cc*. (S+C)

An operand is duplicated.

aa...aa: Operand name

bb...bb: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

cc...cc: Line resulting in the error

S: Stops processing.

[Action]

Check the operand specified in the definition file and correct the error.

KFRB00940-E

Unable to set environment variable, file = *aa...aa*, line = *bb...bb*. (S+C)

The value of an environment variable is invalid.

aa...aa: Definition filename

In the case of the extraction system definition file, extraction environment

definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

bb...bb: Line resulting in the error

S: Stops processing.

[Action]

Check the environment variable specified in the definition file and correct the error.

KFRB00941-E

I/O error occurred, file = *aa...aa*. (S+C)

An I/O error occurred while reading a file required for definition analysis.

aa...aa: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

S: Stops processing.

[Action]

Check the contents of the definition file and correct the error.

KFRB00942-E

Unable to open definition file, definition file = *aa...aa*, errno = *xx...xx*. (S+C)

Datareplicator was unable to open a definition file.

aa...aa: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

xx...xx: Error number set in errno

If the error number is 0, the pathname is invalid.

S: Stops processing.

[Action]

Check the contents of the definition file and correct the error.

KFRB00943-E

Record length exceeds the limit, file = *aa...aa*, line = *bb...bb*. (S+C)

A record length exceeds 80 bytes in a definition file.

aa...aa: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

bb...bb: Line resulting in the error

S: Cancels processing.

[Action]

Check the contents of the definition file and correct the error.

KFRB00944-E

Variable name specified wrong, file = *aa...aa*, line = *bb...bb*. (S+C)

The specified variable contains one of the following errors:

- The variable name is missing
- The variable name is invalid

aa...aa: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

bb...bb: Line resulting in the error

S: Cancels processing.

[Action]

Check the contents of the definition file and correct the error.

KFRB00945-E

Unable to open definition file specified with include, file = *aa...aa*, line = *bb...bb*, definition file = *cc...cc*, errno = *xx...xx*. (S+C)

Datareplicator was unable to open the file specified in include.

aa...aa: Filename

bb...bb: Line resulting in the error

cc...cc: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

xx...xx: Error number set in `errno`

S: Cancels processing.

[Action]

Check the contents of the definition file and correct the error.

KFRB00946-E

Definition *aa...aa* described wrong, file = *bb...bb*, line = *cc...cc*.
(S+C)

An operand argument is invalid.

aa...aa: Operand name

bb...bb: Definition filename

In the case of the extraction system definition file, extraction environment definition file, or a transmission environment definition file, the message displays the name of the work file that is created temporarily during analysis (*definition-filename_tmp*).

cc...cc: Line resulting in the error

S: Cancels processing.

[Action]

Check the contents of the definition file and correct the error.

KFRB00947-E

Command *aa...aa* invalid. (S+C)

A command name is invalid.

aa...aa: Command name

S: Cancels processing.

[Action]

Check the contents of the definition file and correct the error.

KFRB00948-E

Command name in *aa...aa* command invalid. (S+C)

An command name is invalid.

aa...aa: Command name

S: Cancels processing.

[Action]

Check the contents of the definition file and correct the error.

KFRB00949-E

Command argument invalid, command = *aa...aa*. (S+C)

A command argument is invalid.

aa...aa: Command name

S: Cancels processing.

[Action]

Check the contents of the definition file and correct the error.

KFRB00950-E

Operand argument invalid, operand = *aa...aa*. (S+C)

An operand argument is invalid.

aa...aa: Operand name

S: Cancels processing.

[Action]

Check the contents of the definition file and correct the error.

KFRB00951-E

Operand specified twice, command = *aa...aa*. (S+C)

An operand is duplicated.

aa...aa: Command name

S: Cancels processing.

[Action]

Check the contents of the definition file and correct the error.

KFRB00955-E

Specified file size exceeds a size of Character special file,
file name = *aa...aa*, file size = *bb...bb*, specified size = *cc...cc*. (S)

The file size specified in an operand exceeds the actual size of the character special file.

aa...aa: File name

bb...bb: Size of the character special file

cc...cc: File size specified in the operand

S: Stops processing.

[Action] For the file resulting in an error, either correct the corresponding operand value or change the size of the character special file, and then re-execute the command.

KFRB01000-E

<extract definition> expansion error or <reflect definition> parsing error. (C)

Expansion of the extraction definition or analysis of the import definition failed.

Detail information: Displays the line number resulting in the error.

If the KFRB01203-E message was output previously:

If the `check` and `with lock` options are both not specified in the `to` clause of the `load` statement, the value obtained by adding 1 to the line number resulting in the error is displayed.

S: Cancels import processing.

O: Correct the contents of the import definition file, and then restart import processing.

KFRB01001-E

Information for update queue file is not <extract definition>. (C)

The information read from the import information queue file is not the extraction definition.

S: Cancels import processing.

O: This is an internal error. Contact the customer support center.

KFRB01002-E

Insufficient shared memory for parse. (C)

Shortage of shared memory for storing definition analysis results occurred.

S: Cancels import processing.

O: Increase the `defshmsize` operand value in the import environment definition, and then restart import processing.

KFRB01003-E

<extract definition> could not convert to target database language. (C)

Datareplicator was unable to convert the extraction definition to the character codes used for the target database.

S: Cancels import processing.

O: Check the character code set environment settings for any error.

KFRB01004-E

<reflect definition> file not specified. (C)

No import definition file is specified. When the value of the `defmerge` operand in the import environment definition is `false`, you cannot omit the `reffile` operand.

S: Cancels import processing.

O: Create the import definition file and specify its location in the `reffile` operand.

KFRB01005-E

<load statement> not specified in <reflect definition> file. (C)

There is no load statement in the import definition file. When the value of the `defmerge` operand is `false`, at least one load statement is required.

S: Cancels import processing.

O: Specify a load statement in the import definition file.

KFRB01100-E

<extracted data name> in <format clause> in <format statement> is invalid. (C)

The update information name in the format statement's format clause is invalid.

S: Cancels import processing.

O: Correct the update information name and restart import processing.

KFRB01101-E

<extracted data name> in <format clause> in <format statement> is duplicate, <extracted data name> = *aa...aa*. (C)

The update information name is duplicated in the format statement's format clause.

aa...aa: Update information name

S: Cancels import processing.

O: Eliminate the duplicated update information name and restart import processing.

KFRB01102-E

<extracted data name> *aa...aa* in <format clause> in <format statement> is not specified in <extract definition>. (C)

Update information name *aa...aa* specified in the format statement's format clause is undefined in the extraction definition.

aa...aa: Update information name

S: Cancels import processing.

O: Specify the update information name defined in the extraction definition and restart import processing.

KFRB01103-E

<field name> in <name clause> in <format statement> is invalid.
(C)

A field name in the `format statement's name clause` is invalid.

S: Cancels import processing.

O: Correct the field name and restart import processing.

KFRB01104-E

<field name> in <name clause> in <format statement> is
duplicate, <field name> = *aa...aa*. (C)

A field name in the `format statement's name clause` is duplicated.

S: Cancels import processing.

O: Eliminate the duplicated field name and restart import processing.

KFRB01105-E

<field name> in <name clause> in <format statement> exceeds
maximum value. (C)

The number of fields specified in the `format statement's name clause` exceeded the
maximum value.

S: Cancels import processing.

O: Reduce the number of fields and restart import processing.

KFRB01106-E

Number of <extracted data> fields is not equal to number of <name
clause>. (C)

The number of update information fields does not match the number of `name clauses`.

S: Cancels import processing.

O: Correct the specification so that the number of update information fields matches
the number of `name clauses` (excluding a `name clause` with the `const clause`
specified), and then restart import processing. This message might also be displayed
when the field name specified in a `name clause` is invalid. If the number of the update
information fields matches the number of `name clauses`, check the field names
specified in the `name clauses` for errors.

KFRB01107-E

Initial value in <const clause> in <format statement> is
invalid. (C)

The initial value in the `format statement's const clause` is invalid.

S: Cancels import processing.

O: Correct the initial value and restart import processing.

KFRB01108-E

Number of <format statement> exceeds maximum value. (C)

The number of format statements specified in the import definition file exceeded the maximum value.

S: Cancels import processing.

O: Delete excess format statements so that the number of format statements does not exceed the maximum value in the import definition file.

KFRB01109-E

<column data UOC identifier> in <by clause> in <format statement> is invalid. (C)

The identifier of the column data editing UOC routine in the format statement's by clause is invalid.

S: Cancels import processing.

O: Correct the identifier of the column data editing UOC routine and restart import processing.

KFRB01110-E

<nocodecnv> is specified for field that is mapping key column. (C)

The nocodecnv option is specified for a mapping key field.

S: Cancels import processing.

O: Check the fields subject to the nocodecnv option specification, correct the error, and then restart import processing.

KFRB01111-E

Column data UOC is specified for field that is *aa...aa* type extraction column. (C)

The column data editing UOC routine is specified for a field that is an *aa...aa*-type extraction column.

aa...aa: Data type name

BLOB: BLOB type

BINARY: BINARY type

S: Cancels import processing.

O: Check the fields subject to the column data editing UOC routine specification, correct the error, and then restart import processing.

KFRB01112-E

Column data UOC is specified for field that is mapping key column. (C)

A column data editing UOC routine is specified for a mapping key field.

S: Cancels import processing.

O: Check the fields subject to the column data editing UOC routine specification, correct the error, and then restart import processing.

KFRB01113-E

<nocodecnv> is specified for field that is not character column.
(C)

The nocodecnv option is specified for a non-character column field.

S: Cancels import processing.

O: Check the fields subject to the nocodecnv option specification, correct the error, and then restart import processing.

KFRB01114-E

Column data UOC is specified for field that is Abstract Data Type extraction column. (C)

A column data editing UOC routine is specified for extraction of a field that is an abstract data type column.

S: Cancels import processing.

O: Check the fields subject to the column data editing UOC routine specification, correct the error, and then restart import processing.

KFRB01115-E

Unknown Abstract Data Type found, <type name> = aa...aa. (C)

There is an extraction definition for an unsupported abstract data type.

aa...aa: Name of the abstract data type

S: Starts termination processing.

O: Determine whether the specified abstract data type is supported by Datareplicator by referencing *1.4.2 Data linkage for a table using an abstract data type*.

For an abstract data type that is not supported by Datareplicator:

Search the data dictionary table to identify the column of the unsupported abstract data type. Delete that column from the extraction definition, and then initialize the source and target Datareplicators.

For details about how to search the data dictionary table, see the *HiRDB Version 9 UAP Development Guide*.

For an abstract data type that is supported by Datareplicator:

Upgrade HiRDB and Datareplicator to a version that supports the abstract data type. For notes about upgrading, see *Release Notes*.

KFRB01200-E

<field name> in <load clause> in <load statement> is invalid. (C)

A field name in the load statement's load clause is invalid.

S: Cancels import processing.

O: Correct the field name and restart import processing.

KFRB01201-E

<field name> *aa...aa* in <load clause> in <load statement> is not specified in <format statement>. (C)

Field name *aa...aa* specified in the load statement's load clause is not defined in the format statement.

aa...aa: Field name

S: Cancels import processing.

O: Specify a field name defined in the `format` statement, and then restart import processing.

KFRB01202-E

<field name> in <load clause> in <load statement> exceeds maximum value. (C)

The number of field names specified in the load statement's load clause exceeded the maximum value.

S: Cancels import processing.

O: Reduce the number of field names, and then restart import processing.

KFRB01203-E

Number of <field name> in <load clause> in <load statement> is not equal to <reflect table> columns. (C)

The number of field names specified in the load statement's load clause does not match the number of columns in the target table.

S: Cancels import processing.

O: Correct the specification so that the number of field names matches the number of columns in the target table, and then restart import processing.

KFRB01204-E

<extracted data name> in <from clause> in <load statement> is invalid. (C)

The update information name in the load statement's `from` clause is invalid.

S: Cancels import processing.

O: Correct the update information name and restart import processing.

KFRB01205-E

<extracted data name> *aa...aa* in <from clause> in <load statement> is not specified in <extract definition>. (C)

Update information name *aa...aa* specified in the load statement's from clause is undefined in the extraction definition.

aa...aa: Update information name

S: Cancels import processing.

O: Specify an update information name that is defined in the extraction definition and restart import processing.

KFRB01206-E

<reflect table> existence check failed, <table name> = *aa...aa*. (C)

Checking for a table subject to import processing failed.

aa...aa: Table name

Detail information: Displays the HiRDB message indicating the cause of the error.

S: Cancels import processing.

O: Check the SQL code displayed as detail information and eliminate the cause of the error.

KFRB01207-E

<table name> in <to clause> in <load statement> is not real table, <table name> = *aa...aa*. (C)

The table name specified in the load statement's to clause is not a base table.

aa...aa: Table name

S: Cancels import processing.

O: Specify a base table, and then restart import processing.

KFRB01208-E

<authorization identifier> in <to clause> in <load statement> is invalid. (C)

The authorization identifier in the load statement's to clause is invalid.

S: Cancels import processing.

O: Correct the authorization identifier and restart import processing.

KFRB01209-E

<table identifier> in <to clause> in <load statement> is invalid. (C)

The table identifier in the load statement's to clause is invalid.

S: Cancels import processing.

O: Correct the table identifier and restart import processing.

KFRB01210-E

<uoc name> in <by clause> in <load statement> is invalid. (C)

The UOC routine name in the load statement's by clause is invalid.

S: Cancels import processing.

O: Correct the UOC routine name and restart import processing.

KFRB01211-E

Not specified all <mapping key> field in <load clause> in <load statement>. (C)

Some mapping key fields are missing in the load statement's load clause.

S: Cancels import processing.

O: Add the missing mapping key fields and restart import processing.

KFRB01212-E

Duplicate <mapping key> field in <load clause> in <load statement>. (C)

A mapping key field is duplicated in the load statement's load clause.

S: Cancels import processing.

O: Eliminate the duplicated mapping key field and restart import processing.

KFRB01213-E

Number of <load statement> exceeds maximum value. (C)

The number of load statements specified in the import definition file exceeded the maximum.

S: Cancels import processing.

O: Correct the specification so that the number of load statements in the import definition file, including tables subject to implicit import processing (target tables for which the source table name is assumed because there is no explicitly specified load statement), does not exceed the maximum.

KFRB01214-E

Unable to specified <constant field>. (C)

Specification of a definition field is prohibited.

When you use a UOC routine, you cannot specify a constant field in the load clause.

S: Cancels import processing.

O: Eliminate the constant field from the load clause and restart import processing.

KFRB01215-E

Repetition columns cannot be imported to a time-ordered information table, extract data name = *aa...aa*, field no = *bb...bb*. (C)

Datareplicator cannot import a repetition column into a time-ordered table.

aa...aa: Update information name

bb...bb: Sequence number assigned to the field name specified in the `load` clause (this information is displayed only once, even though multiple repetition-column fields are specified)

S: Cancels import processing.

O: Either delete the field name for the repetition column from the `load` clause or delete `timestamp` from the `to` clause, and then restart import processing.

KFRB01216-E

Failed to get information of unique check of mapping key, SQLCODE = *aa...aa*. (C)

Datareplicator failed to obtain the information that is needed for unique check of mapping key column.

aa...aa: SQLCODE

S: Cancels processing.

O: See the detail information, eliminate the cause of the error, and then restart import processing.

KFRB01217-E

Mapping key column does not satisfy condition of unique check, table name = *aa...aa*, code = *bb...bb*. (C)

The mapping key column does not satisfy the conditions of the unique check.

aa...aa: Target table name

bb...bb: Cause code

UNIQUE: An index satisfying the conditions of the unique check has not been defined for the target table.

NOT NULL: The index satisfying the conditions of the unique check includes a column whose attribute is not NOT NULL.

S: Cancels processing.

O: Check the mapping key column for errors. If there is an error, correct the definition. If there is no error in the specification, take appropriate action on the basis of the cause code. If you cannot take such action, change the value of `mapping_key_check` in the import environment definition or the value of the check clause in the import definition,

and then re-execute import processing.

- UNIQUE

For the target table, define the index that satisfies the conditions of the unique check.

- NOT NULL

Change the component column of the index satisfying the conditions of the unique check to NOT NULL attribute.

KFRB01300-E

<group name> in <group clause> in <group statement> is invalid.
(C)

A group name in the group statement's group clause is invalid.

S: Cancels import processing.

O: Correct the group name and restart import processing.

KFRB01301-E

<group name> in <group clause> in <group statement> is
duplicate, <group name> = aa...aa. (C)

A group name is duplicated in the group statement's group clause.

aa...aa: Group name

S: Cancels import processing.

O: Eliminate the duplicated group name, and then restart import processing.

KFRB01302-E

<authorization identifier> in <by clause> in <group statement>
is invalid. (C)

The authorization identifier in the group statement's by clause is invalid.

S: Cancels import processing.

O: Correct the authorization identifier and restart import processing.

KFRB01303-E

<table identifier> in <by clause> in <group statement> is
invalid. (C)

The table identifier in the group statement's by clause is invalid.

S: Cancels import processing.

O: Correct the table identifier and restart import processing.

KFRB01304-E

<table name> in <by clause> in <group statement> is not <reflect table>, <table name> = *aa...aa*. (C)

The table specified in the group statement's by clause is not subject to import processing.

aa...aa: Table name

S: Cancels import processing.

O: Specify a table subject to import processing and restart import processing.

KFRB01305-E

<group name> in <group clause> in <group statement> exceeds maximum value. (C)

The number of group names specified in the group statement's group clause exceeded the maximum.

S: Cancels import processing.

O: Reduce the number of group names and restart import processing.

KFRB01306-E

<table name> in <by clause> in <group statement> is duplicate, <table name> = *aa...aa*. (C)

A table name is duplicated in the group statement's by clause.

aa...aa: Table name

S: Cancels import processing.

O: Eliminate the duplicated table name and restart import processing.

KFRB01307-E

Column is not mapping key in <having clause> or <hash-divide key>, <column name> = *aa...aa*. (C)

A column name or hash partitioning column specified in the group statement's having clause is not a mapping key.

aa...aa: Column name

S: Cancels import processing.

O: Specify the mapping key for the column in the conditional statement, and then restart import processing. If the hash partitioning method is being used, match the mapping key and the hash partitioning key, and then restart import processing.

KFRB01308-E

Column name is not exist in <having clause>, <column name> = *aa...aa*. (C)

A nonexistent column is specified in the `group` statement's `having` clause.

aa...aa: Column name

S: Cancels import processing.

O: Specify a column from the table in the conditional statement and restart import processing.

KFRB01309-E

Column type and constant type is not matched in `<having clause>`, `<column name> = aa...aa`. (C)

The data types do not match between the constant and column specified in the `group` statement's `having` clause.

aa...aa: Column name

S: Cancels import processing.

O: Correct the conditional statement and restart import processing.

KFRB01310-E

Number of key range overflow in `<group clause>`, `<group name> = aa...aa`. (C)

The number of key ranges specified in the `group` statement exceeded the maximum.

aa...aa: Import group name

S: Cancels import processing.

O: Correct the specification so that the number of key ranges for the group does not exceed the maximum, and then restart import processing.

KFRB01311-E

Number of range condition overflow in `<having clause>`, `<group name> = aa...aa`. (C)

The number of conditional statements specified in the `group` statement's `having` clause exceeded the maximum.

aa...aa: Import group name

S: Cancels import processing.

O: Correct the specification so that the number of conditional statements for the key range does not exceed the maximum, and then restart import processing.

KFRB01312-E

Decimal data overflow in `<having clause>`, `<decimal data> = aa...aa`. (C)

The decimal data value specified in the `group` statement's `having` clause resulted in an overflow.

aa...aa: Decimal data value

S: Cancels import processing.

O: Correct the decimal data in the conditional statement and restart import processing.

KFRB01313-E

Invalid data format in `<having clause>`, `<constant data> = aa...aa.` (C)

A constant value in the `group` statement's `having` clause is invalid.

aa...aa: Value of the constant

S: Cancels import processing.

O: Correct the constant in the conditional statement and restart import processing.

KFRB01314-E

Invalid date format in `<having clause>`, `<date data> = aa...aa.` (C)

The date data specified in the external format is invalid in the `group` statement's `having` clause.

aa...aa: Date data

S: Cancels import processing.

O: Correct the date data in the conditional statement and restart import processing.

KFRB01315-E

Invalid time format in `<having clause>`, `<time data> = aa...aa.` (C)

The time data specified in the external format is invalid in the `group` statement's `having` clause.

aa...aa: Time data

S: Cancels import processing.

O: Correct the time data in the conditional statement and restart import processing.

KFRB01316-E

Invalid FES name in `<in clause>`, `<FES name> = aa...aa.` (C)

The target FES in the `group` statement's `in` clause is invalid.

aa...aa: Server name of the target FES

S: Cancels import processing.

O: Correct the server name of the target FES and restart import processing.

KFRB01317-E

Invalid FES number in `<in clause>`, `<FES number> = aa...aa.` (C)

The number of target FESs in the `group` statement's `in` clause is invalid.

aa...aa: Number of target FESs

S: Cancels import processing.

O: Correct the number of target FESs and restart import processing.

KFRB01318-E

Not exist hash function library, library name = *aa...aa*. (C)

A hash function library does not exist.

aa...aa: Library name

S: Cancels import processing.

O: Install the HiRDB hash function library and restart import processing.

KFRB01319-E

Failed to load hash function library, library name = *aa...aa*, errno = *bb...bb*. (C)

A load error occurred in the hash function library.

aa...aa: Library name

bb...bb: Error number

S: Cancels import processing.

O: See `errno.h` or the OS documentation for details about the error number, eliminate the cause of the error, and then restart import processing.

KFRB01320-E

Table is not hash-divided in <group clause>, <table name> = *aa...aa*. (C)

A table specified in the `group` statement has not been partitioned by the hash partitioning method.

aa...aa: Table name

S: Cancels import processing.

O: Modify the target table definition and restart import processing.

KFRB01321-E

Not support hash function in <group clause>, <table name> = *aa...aa*. (C)

Hash partitioning is not supported for a table specified in the `group` statement.

aa...aa: Table name

S: Cancels import processing.

O: Modify the table definition and restart import processing.

KFRB01322-E

Failed to get hash-divided table information , <table name> = *aa...aa*. (C)

Datareplicator was unable to obtain information about a hash-partitioned table.

aa...aa: Table name

S: Cancels import processing.

O: See the detail information output to the error file, eliminate the cause of the error, and then restart import processing.

KFRB01323-E

Hash key column type mapping error, <column name> = *aa...aa*. (C)

A hash key column's data type does not match the data type at the source.

aa...aa: Column name

S: Cancels import processing.

O: Modify the target table definition and restart import processing.

KFRB01324-E

Invalid divide number in <divide clause>, <divide number> = *aa...aa*. (C)

The splits count in the group statement's divide clause is invalid.

aa...aa: Splits count

S: Cancels import processing.

O: Correct the splits count for hash partitioning and restart import processing.

KFRB01325-E

Decimal data overflow in <divide clause>, <decimal data> = *aa...aa*. (C)

The decimal data value specified in the group statement's divide clause resulted in an overflow.

aa...aa: Decimal data value

S: Cancels import processing.

O: Correct the splits count for hash partitioning and restart import processing.

KFRB01326-E

<rdarea name> in <rdarea_list clause> in <group statement> is invalid. (C)

An invalid RDAREA name was specified in the hash partitioning definition in the group statement.

S: Cancels import processing.

O: Correct the RDAREA name, and then restart import processing.

KFRB01327-E

<rdarea name> in <rdarea_list clause> in <group statement> is duplicate, <rdarea name> = *aa...aa*. (C)

A duplicated RDAREA name was specified in the hash partitioning definition in the group statement. RDAREA name = *aa...aa*

aa...aa: RDAREA name

S: Cancels import processing.

O: Eliminate the duplicated RDAREA name, and then restart import processing.

KFRB01328-E

FES, which includes RDareas and not, coexist error in <hash_fes_clause>, <table name> = *aa...aa*. (C)

The hash partitioning definition in the group statement contains FES specifications both with and without RDAREAs. Table name = *aa...aa*

aa...aa: Table name

S: Cancels import processing.

O: Correct the specifications in terms of RDAREA names, and then restart import processing.

KFRB01329-E

Invalid timestamp format in <having clause>, <timestamp data> = *aa...aa*. (C)

Invalid external-format time stamp data was specified in the having clause of the group statement.

aa...aa: Time stamp data

S: Cancels processing.

O: Correct the time stamp data in the conditional statement, and then re-execute import processing.

KFRB01401-E

Dual file definition has a mistake, file = *aa...aa*, line = *bb...bb*, code = *cc...cc*. (C + S + L)

There is an error in the duplexing definition.

aa...aa: Duplexing definition file name

bb...bb: Line number

cc...cc: Cause code

S: Cancels processing.

O: Correct the duplexing definition, and then restart the system. For details about the cause code, see *10.4 List of cause codes*.

KFRB01402-E

Unable to use dual file function, code = *aa...aa*, info = *bb...bb*. (C + S + L)

The file duplexing function is not available due to an error.

aa...aa: Cause code

bb...bb: Detail information

S: Cancels processing.

O: Eliminate the cause of the error on the basis of the cause code, and then restart the system. For details about the cause code, see *10.4 List of cause codes*.

KFRB01403-E

Operand combination error, definition file = *aa...aa*, operand = *bb...bb*, definition file = *cc...cc*, operand = *dd...dd*. (C)

A combination of operands is invalid.

aa...aa: Definition filename

bb...bb: Operand name

cc...cc: Definition filename

dd...dd: Operand name

S: Cancels processing.

O: Check the combination of operands, correct the error, and then re-execute the command.

KFRB02001-E

Invalid data of reflect status file. (C)

The contents of the import status file are invalid.

S: Cancels processing.

O: Re-execute using the `hdsstart -i` command.

KFRB02002-E

Socket connection error, errno = *xx...xx*. (C)

A socket connection error occurred.

xx...xx: Error number set in errno

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB02003-E

Port check error, code = yy, zz...zz. (C)

Port checking resulted in an error.

yy: Detail code

zz...zz: information for each detail code (zz...z is displayed only when yy is 3, 4, 5, 7, 11, 12, 13, 14, or 15)

S: Cancels processing.

O: Take appropriate action, as indicated in the table below:

Detail code	Description	Action
1	Attempt was made to establish connection with the wrong source system.	Establish connection with the correct source system.
	The target system's reception process (or thread) has not been generated.	Eliminate the cause of the error and reestablish connection.
2	The source and target systems are not compatible for purposes of connection establishment.	Check the combination of the source and target systems and establish connection with the correct combination.
3	Specified data linkage identifier is not defined in the target Datareplicator's import system definition.	Add the data linkage identifier specified in the source system to the import system definition, and then reestablish connection.
4	The target Datareplicator identifier specified in the target Datareplicator's import system definition does not match the specification at the source system.	Correct the target Datareplicator identifier specified at the source system, and then reestablish connection.
5	Attempt was made to send update information that was not a continuation of the previously transmitted update information.	Contact the source system's customer engineer.
6	The source database is not supported by the target Datareplicator.	Contact the source system's customer engineer.
7	The target Datareplicator cannot allocate a reception buffer because the information buffer at the source is too large.	Reduce the source system's update information buffer size.
8	Attempt was made to send update information using an unsupported character code set.	Contact the source system's customer engineer.

Detail code	Description	Action
9	The specified send data compression method is not supported.	Contact the source system's customer engineer.
10	Attempt was made to send update information in an unsupported format.	Contact the source system's customer engineer.
11	The specified data linkage identifier is already being used by another source system.	Correct the data linkage identifier specified at the source system, and then reestablish connection.
	The target system has been restarted.	Restart the corresponding destination at the source system.
12	I/O error occurred in the import status file during start processing.	Eliminate the cause of the file error and restart the target Datareplicator.
13	Semaphore lock or unlock error occurred during start processing.	Correct the semaphore error and restart the target Datareplicator.
14	I/O error occurred in the import information queue file during start processing.	Eliminate the cause of the file error and restart the target Datareplicator.
15	Processing cannot continue because the source system's update information buffer size has changed since the previous session.	Restore the previous update information buffer size, and then reestablish connection.
16	Received data is not port check information.	Check if any application is communicating illegally with the service specified in the <code>hdsservice</code> operand in the import environment definition.
17	Processing cannot continue because the type of database has changed since the previous session.	Contact the source system's customer engineer.
18	Combination of character codes is invalid.	Check the <code>dblocale</code> operand in the import system definition and in the extraction system definition for any error.
19	Update information was sent for the data linkage identifier that was used for reading update information.	Specify <code>rcv</code> in the <code>ujcodekind</code> operand in the import environment definition.
20	An invalid DBMS was specified in Datareplicator Extension's <code>dbkind</code> operand as being subject to import processing.	Correct the specification of Datareplicator Extension's <code>dbkind</code> operand, and then re-execute the command.
	A related program [#] required for linking data between a mainframe database and Oracle or SQLServer has not been installed.	Install the required related program, [#] and then re-execute the command. If SAM files are used, install the related program, re-create the SAM files, and then re-execute the command.

Detail code	Description	Action
21	HiRDB Datareplicator Extension has not been installed.	Install the HiRDB Datareplicator Extension and re-execute.
22	The other company's database connection license has expired.	Contact the system administrator.
23	The license file is damaged.	Reinstall the HiRDB Datareplicator Extension.
24	The connection request that was received from the source system does not support repetition columns.	Upgrade the source system's Datareplicator so that it supports repetition columns; or, eliminate repetition columns from the extraction definition.
25	The <code>dblocale</code> operands do not match between the extraction system definition and the import system definition.	Specify the same character code system in the <code>dblocale</code> operands in the extraction system definition and the import system definition, and then re-execute the command.
26	The <code>syncgroup001</code> operand is specified in the import system definition, but the source system is set to not send transaction branch information.	See <i>3.7.2 Preparations for the import transaction synchronization facility</i> to check the prerequisites for using the import transaction synchronization facility.
27	The synchronous event code that was sent does not match the previous synchronous event code that was sent.	Take one of the following actions: <ul style="list-style-type: none"> In the transmission environment definition, reset the <code>eventsync</code> operand to the previous value. Initialize the target Datareplicator.

#

The following shows the related programs required for linking data between a mainframe database and Oracle or SQLServer:

Source system	Oracle	SQLServer
XDM/DS	XDM/DS/extended linkage 1	XDM/DS/extended linkage 2
VOS3 Database Datareplicator	XDM/DS/extended linkage 1	Data cannot be linked.
VOS1 PDMII E2 VOS1 RDB1 E2	VOS1 Database Adaptor 1	VOS1 Database Adaptor 2

KFRB02004-E

Connection was closed. (C)

Communication line was disconnected.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB02005-E

Extracted data is invalid. (C)

Update information is invalid.

S: Cancels processing.

O: This is an internal error. Contact the customer support center.

KFRB02006-E

Received data sequence is invalid. (C)

Received data sequence is invalid.

S: Cancels processing.

O: This is an internal error. Contact the customer support center.

KFRB02007-E

Unable to save extract definition in reflect status file, because reflect status file is insufficient, necessary statssize is xx...xx. (C)

Datareplicator cannot store the extraction definition because the import status file is too small.

xx...xx: Size required for status file (Byte)

S: Cancels processing.

O: Increase the statssize operand value in the import environment definition and restart using the `hdsstart -i` command.

KFRB02008-E

Connection count exceed maximum value. (C)

Attempt was made to establish data linkage using more connections than the maximum.

S: Cancels processing.

O: Check that the data linkage identifier to be used to establish connection is specified in the import system definition. If necessary, correct the specification and re-execute.

KFRB02009-E

Data send error, errno = xx...xx. (C)

An error occurred during data transmission to the source system.

xx...xx: Error number set in errno

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number,

see `errno.h` or the OS documentation.

KFRB02010-E

Data receive error, `errno = xx...xx`. (C)

An error occurred during data reception from the source system.

`xx...xx`: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB02011-E

Receive unknown data. (C)

Unknown data was received.

S: Cancels processing.

O: This is an internal error. Contact the customer support center.

KFRB02012-E

Connection accept error, `errno = xx...xx`. (C)

An error occurred while accepting a connection request.

`xx...xx`: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB02013-E

Socket option set error, `errno = xx...xx`. (C)

An error occurred while setting up the socket option. error number = `xx...xx`

`xx...xx`: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB02014-E

Select system call error, `errno = xx...xx`. (C)

The `select` system call resulted in an error.

`xx...xx`: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number,

see `errno.h` or the OS documentation.

KFRB02015-E

Receive data exceed maximum receive data length. (C)

The data to be received exceeds the maximum size.

S: Cancels processing.

O: This is an internal error. Contact the customer support center.

KFRB02016-E

Receive data extension failed. (C)

Receive data expansion processing failed.

S: Cancels processing.

O: This is an internal error. Contact the customer support center.

KFRB02017-E

Receive data length is 0 byte. (C)

The length of the data to be received is 0 bytes.

S: Cancels processing.

O: This is an internal error. Contact the customer support center.

KFRB02018-E

Specified service name not found in services file, service name = *aa...aa*. (C)

A specified service name was not found in the services file.

aa...aa: Service name

S: Cancels processing.

O: Add this service name to the services file or specify a service name that is in the services file.

KFRB02019-I

Communication master process started, protocol = *aa...aa*. (C)

The import communication master process has started.

aa...aa: Protocol type

S: Resumes processing.

KFRB02020-I

Communication master process ended, protocol = *aa...aa*. (C)

The import communication master process has terminated.

aa...aa: Protocol type

S: Resumes processing.

KFRB02021-I

Extracted data receipt started. (C)

Datareplicator has started receiving update information.

S: Resumes processing.

KFRB02022-I

Extracted data receipt ended. (C)

Reception of update information has been completed.

S: Resumes processing

KFRB02023-I

Accepted stop request in receiver, request kind = *aa...aa*. (C)

Datareplicator has accepted a termination request during reception processing.

aa...aa: Request type:

Normal: Because Datareplicator has accepted the `hdsstop` command, processing will be terminated when the end-of-transmission message is received.

Event: Because Datareplicator has accepted the `hdsstop -t event` command, processing will be terminated when the first event occurs.

Immediate: Because Datareplicator has accepted the `hdsstop -t immediate` command, reception processing will be terminated.

Force: Because Datareplicator has accepted the `hdsstop -t force` command, reception processing will be terminated.

S: Resumes processing.

KFRB02024-E

OSI function call error. function name = *aa...aa*, errno = *xx...xx*. (C)

An OSI function call resulted in an error.

aa...aa: OSI function resulting in the error

xx...xx: Error number returned by the OSI function

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `tiuser.h` or the OS documentation.

KFRB02025-E

OSI communication program is not installed. (C)

The program required for OSI communications is not installed.

S: Cancels processing.

O: Install the program required for OSI communications and restart the target Datareplicator.

KFRB02026-E

OSI communication program is not active. (C)

The program required for OSI communications is not active.

S: Cancels processing.

O: Start the program required for OSI communications, and then restart the target Datareplicator.

KFRB02027-E

Sender process was held. (E)

The transmission process was shut down.

S: Cancels processing.

O: Restore conformity between the source and target databases, and then start the transmission process.

KFRB02028-E

Connect system call error, errno = *xx...xx*, host name = *aa...aa*, service name = *bb...bb*. (E)

A Connect system call resulted in an error.

xx...xx: Error number set in errno

aa...aa: Host name

bb...bb: Service name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `tiuser.h` or the OS documentation.

KFRB02029-E

Missing specified host name in hosts file, host name = *aa...aa*. (E)

The specified host name was not found in the hosts file.

aa...aa: Host name

S: Cancels processing.

O: Specify the correct host name in the `hosts` file and re-execute.

KFRB02030-E

Error occurred on target system, message number = *nnnnn*. (C)

An error occurred at the target system.

nnnnn: Message number of the error that occurred at the target system

nnnnn corresponds to `KFRBnnnnn-T`, the message ID output by Datareplicator.

S: Cancels processing.

O: Correct the error at the target system and re-execute.

KFRB02031-I

Sender process started. (E)

The transmission process has started.

S: Resumes processing.

KFRB02032-I

Sender process ended, status = *aa...aa*. (E)

The transmission process has ended.

aa...aa: Termination status:

Auto-Normal: Normal termination by the termination log

Manual-Normal: Normal termination by command

Event: Termination by event detection

Auto-Force: Forced termination by the termination log

Manual-Force: Forced termination by command

Hold: Termination by reduced mode operation

Error: Termination by error detection

S: Resumes processing.

KFRB02033-I

Sender process sent log data.(transaction (*xx...xx*, *yy...yy*): insert
(*xx...xx*, *yy...yy*): update (*xx...xx*, *yy...yy*): delete (*xx...xx*, *yy...yy*): purge
(*xx...xx*, *yy...yy*)) (E)

Datareplicator has sent update information.

This message displays the number of each type of SQL statements executed, as shown below:

transaction: Number of transactions in the update information

insert: Number of insert statements issued

update: Number of update statements issued

delete: Number of delete statements issued

purge: Number of purge statements issued

xx...xx: Count within this unit of update information

yy...yy: Total in all update information transmitted

S: Resumes processing.

KFRB02034-I

Sender process connected to target system, host name = *aa...aa*, service name = *bb...bb*. (E)

Connection has been established with the target system.

aa...aa: Host name

bb...bb: Service name

S: Resumes processing.

KFRB02035-W

Target system is not active, host name = *aa...aa*, services name = *bb...bb*. (C)

The target system is not active.

aa...aa: Host name

bb...bb: Service name

S: Resumes processing.

O: Start the target system.

KFRB02036-I

Sender process sent event data, event code = *xx...xx*. (E)

Event data has been transmitted.

xx...xx: Event code

S: Resumes processing.

KFRB02037-I

Sender process accepted stop request, request kind = *aa...aa*. (E)

A termination request was accepted during transmission processing.

aa...aa: Request type:

Normal: Normal termination request

S: Resumes processing.

KFRB02038-E

Value of editbufsize operand must be more than *aa...aa*. (E)

The editbufsize operand value must be greater than *aa...aa*.

aa...aa: Required buffer size

S: Cancels processing.

O: Increase the `editbufsize` operand value in the transmission environment definition and re-execute.

KFRB02039-E

No response from target system, host name = *aa...aa*, service name = *bb...bb*. (C)

The target system is not responding.

aa...aa: Host name

bb...bb: Service name

S: Cancels processing.

O: Check whether an error has occurred at the target system.

KFRB02040-I

Connection close request from source site system accepted. (C)

A transmission termination request was received from the source system.

S: Resumes processing.

KFRB02041-W

Halfway end of multi-byte character code found in data, substitute space code. (E)

A multi-byte character is incomplete in the data subject to code conversion; it will be replaced with a space.

S: Resumes processing.

KFRB02042-E

Log data is inconsistency with extract definition. (C)

The extracted update information contradicts the extraction definition.

The extraction definition might have been modified before all update information in the extraction information queue file was transmitted.

S: Cancels processing.

O: Before restarting data linkage, synchronize the data linkage environments at the source and target, initialize them, and then re-create the target database on the basis of the source database.

KFRB02043-E

Insufficient buffer occurred while editing log data. (E)

A shortage occurred in the update information buffer during the editing of update information.

S: Cancels processing.

O: Specify an appropriate buffer size based on the formula for determining the size of the update information editing buffer, and then re-execute.

KFRB02046-I

Sender process ended. (E)

The transmission process has ended.

S: Terminates processing.

KFRB02047-E

Target system is stopping now. (E)

The target system is terminating.

S: Terminates processing.

O: Restart the target Datareplicator. The source Datareplicator will retry as many time as the retries count specified in the transmission environment definition. If the transmission process (transmission processing) has terminated, restart it.

KFRB02048-I

Sender process connected to target system, NSAP address = *aa...aa*, Tselector = *bb...bb*. (E)

Connection was established with the target system.

aa...aa: NSAP address

bb...bb: T-selector value

S: Resumes processing.

KFRB02049-W

Target system is not active, NSAP address = *aa...aa*, Tselector = *bb...bb*. (E)

The target system is not active.

aa...aa: NSAP address

bb...bb: T-selector value

S: Resumes processing.

O: Start the target system.

KFRB02050-E

No response from target system, NSAP address = *aa...aa*, Tselector = *bb...bb*. (E)

The target system is not responding.

aa...aa: NSAP address

bb...bb: T-selector value

S: Cancels processing.

O: Check for an error at the target system.

KFRB02051-E

Fail to convert address for osi. (E)

An error occurred while converting the address to be used for OSI communications.

S: Cancels processing.

O: Check for an error in the NSAP address or the T-selector value.

KFRB02052-E

Error occurred in send-uoc, status = xx...xx. (E)

An error occurred in the send UOC routine.

xx...xx: Status code set by the UOC routine

S: Cancels processing.

O: Check the status code and take appropriate action.

KFRB02053-W

Connection was closed. Sender process retry connect to target system, reason = aa...aa. (E)

Line was disconnected. Datareplicator will reestablish the connection.

aa...aa: Reason for the line disconnection:

Close: A close packet was detected.

Econnreset: Termination of the data reception process was detected at the target system while waiting for a response.

Epipe: Termination of the data reception process was detected at the target system during data transmission.

Que-full: Queue-full status was detected at the target system.

other: Other error was detected.

S: Reestablishes the line connection and resumes processing.

KFRB02054-E

No received data in specified time. (C)

No data was transmitted within the specified time.

S: Cancels processing.

O: Check for an error at the target system. If this message was output at the source system, restart the corresponding destination.

KFRB02055-E

No sent data in specified time. (E)

Datareplicator was unable to send data within the specified time.

S: Cancels processing.

O: Check for an error at the target system. If this message was output at the source system, restart the corresponding destination.

KFRB02056-E

In spite of reach maximum retry times, unable to connect to target system, host name = *aa...aa*, service name = *bb...bb*, reason = *cc...cc*. (E)

Datareplicator was unable to establish connection with the target system within the specified number of retries.

aa...aa: Host name

bb...bb: Service name

cc...cc: Error number set in errno

S: Cancels processing.

O: Start the target system and restart the corresponding destination.

KFRB02057-E

In spite of reach maximum retry times, unable to connect to target system, NSAP address = *aa...aa*, T-selector = *bb...bb*, reason = *cc...cc*. (E)

Datareplicator was unable to establish connection with the target system within the specified number of retries.

aa...aa: NSAP address

bb...bb: T-selector value

cc...cc: Error number returned by the OSI function

S: Cancels processing.

O: Start the target system and restart the corresponding destination.

KFRB02058-W

Log data is PURGE TABLE of partitioned table. This log data is no out. table name = *aa...aa*. (E)

Update information for a partitioned table is PURGE TABLE; Datareplicator will ignore this update information.

aa...aa: Table name

S: Resumes processing.

O: Because conformity between the source and target databases has been lost, synchronize the target database with the source database.

KFRB02059-W

Log data is PURGE TABLE of partitioned table. This log data change to event data. event code = *aa...aa*, table name = *bb...bb*. (E)

Update information for a partitioned table is PURGE TABLE; Datareplicator will change it to event data.

aa...aa: Event code

bb...bb: Table name

S: Resumes processing.

KFRB02060-W

Undetermined transaction deleted, count = *xx...xx*. (E)

Unresolved transactions are being deleted.

xx...xx: Number of unresolved transactions being deleted

S: Resumes processing.

KFRB02061-E

UOC for obtain ADT input data returned error status, UOC name = *aa...aa*, information = *bb...bb*. (E)

Error-level status was returned from the abstract data type input data acquisition UOC routine.

aa...aa: UOC routine name

bb...bb: Additional information

S: Cancels processing.

O: Check the additional information, resolve the problem, and then restart the corresponding destination.

KFRB02062-W

UOC for obtain ADT input data returned warning status, UOC name = *aa...aa*, information = *bb...bb*. (E)

Warning-level status was returned from the abstract data type input data acquisition UOC routine.

aa...aa: UOC routine name

bb...bb: Additional information

S: Resumes processing.

KFRB02063-W

Cannot accept restart event while processing restart event,
rejected event = *aa...aa*. (C)

Datareplicator cannot accept a restart event because restart event processing is already underway.

aa...aa: Event code

S: Resumes processing.

KFRB02064-W

Stop request accepted from other process. (C)

A termination request from another process was accepted.

S: Starts termination processing.

KFRB02065-W

Unable to start data transmission for <send identifier: *aa...aa*>,
because reduction status is continued. (E)

Datareplicator cannot start transmission processing for target identifier *aa...aa* because reduced mode operation is in effect.

aa...aa: Target identifier

S: Cancels transmission to this destination.

O: To restart transmission processing, cancel the reduced mode operation and restart this destination.

KFRB02066-W

DSID was not synchronized reflection group in target system. (C)

The `reflect_mode` operand in the transmission environment definition is invalid for one of the following reasons:

- In the target system, the data linkage identifiers are not defined as a synchronous import group.
- The target system does not support synchronous import groups.

S: Resumes processing without using the import transaction synchronization facility.

O: To use the import transaction synchronization facility, define the data linkage identifiers as a synchronous import group in the target system. If the target system does not support synchronous import groups, upgrade it to a version that supports synchronous import groups, and then define the synchronous import group

KFRB02072-E

Invalid extract data format for target system, extract data
format kind = *aa...aa*, table name = *bb...bb*. (C)

An invalid update information format that is not supported by the source system was

detected.

aa...aa: Type of update information format

- 1: Backward deletion update log format of BLOB or BINARY type when the SUBSTR scalar function is used

bb...bb: Name of table subject to extraction

The mapping key value is displayed as additional information.

S: Resumes processing.

[Action]

There is an inconsistency between the source and target databases in the BLOB-type or BINARY-type data using the SUBSTR scalar function (backward deletion updating) on the row with the mapping key value displayed as additional information. Restore consistency as follows:

1. Terminate updating applications at the source database and apply all extracted update information to the target database.
2. Terminate the target Datareplicator normally.
3. Upgrade the target Datareplicator to a version (08-01 or later) that can link BLOB-type or BINARY-type backward deletion updating data.

If the target Datareplicator cannot be upgraded to version 08-01 or later, terminate HiRDB normally, and then delete BACKWARD_CUTOFF_UPDATE from the pd_rpl_func_control operand in HiRDB's system common definition. Then start HiRDB normally.

4. Identify the row in the source and target databases that contains the inconsistency on the basis of the mapping key value displayed in this message as additional information.
5. Import the row identified in step 4 from the source database to the target database to restore consistency.
6. Start the target Datareplicator normally.

KFRB02080-W

Detected invalid extract data, extract data kind = *aa...aa*, SQL kind = *bb...bb*, table name = *cc...cc*. (C)

Unsupported update information was detected.

aa...aa: Type of update information format

- 1: A table for which the WITHOUT ROLLBACK option is specified was updated by a statement other than UPDATE.

bb...bb: SQL type

cc...cc: Name of the table subject to extraction

The mapping key value is displayed as additional information.

S: Resumes processing.

[Action] If *aa...aa* is 1, there is an inconsistency between the source and target databases. Restore consistency as follows:

1. Terminate updating applications at the source database and apply all extracted update information.
2. Terminate the target Datareplicator normally.
3. Identify the row in the source and target databases that contains the inconsistency on the basis of the mapping key value displayed in this message as additional information.
4. Import the row identified in step 3 from the source database to the target database to synchronize the source and target databases.
5. Start the target system normally.

KFRB02500-I

Send-master process started. (E)

The transmission master process has started.

S: Resumes processing.

KFRB02501-I

Send-master process ended, status = *aa...aa*. (E)

The transmission master process has terminated.

aa...aa: Termination status:

Normal: Normal termination

No-Process: Terminated because there was no available transmission process

Signal: Termination by signal reception

Error: Termination by error detection

S: Terminates processing.

KFRB02502-I

Data transmission for <send identifier: *aa...aa*> ended, status = *bb...bb*. (E)

Transmission processing for target definition *aa...aa* was terminated.

aa...aa: Target identifier

bb...bb: Termination status:

Auto-Normal: Normal termination by termination log detection

Manual-Normal: Normal termination by command

Event: Termination by event detection

Auto-Force: Forced termination by termination log detection

Manual-Force: Forced termination by command

Hold: Termination by reduced mode operation

Error: Termination by error detection

S: Cancels the corresponding transmission processing.

KFRB02506-E

Found abnormal termination of sender process. (E)

Abnormal termination of transmission processing was detected.

S: Cancels the transmission process resulting in the error.

O: This is an internal error. Check the error message that was displayed immediately before this message. If no error message was displayed, contact the customer support center.

KFRB02507-E

Unable to begin data transmission for <send identifier : aa...aa>
(E)

Datareplicator is unable to start transmission processing for target identifier aa...aa.

aa...aa: Target identifier

S: Cancels processing.

O: Eliminate the cause of the error indicated in the message output immediately previously.

KFRB02508-I

Transmission started for <send identifier : aa...a>. (E)

Transmission processing for target identifier aa...aa has begun.

aa...aa: Target identifier

S: Resumes processing.

KFRB02601-W

Warning from SQL output function, info = "aa...aa". (E)

A warning was detected during update-SQL output processing. If multiple warning errors were detected from a single row of SQL output information, only the message for the last detected error is displayed.

aa...aa: Detail information

Message	Description
"because a data conversion error occurred, the value is printed with hexadecimal, datatype = 0xaa...aa."	An error occurred during conversion from packed data to character string. <i>aa...aa</i> : Data code [#]
"because of unsupported data type, ' <i>aa...aa</i> ' was printed instead of the value, datatype = 0xbb...bb."	There is an unsupported column data type. <i>aa...aa</i> : Column data type *BLOB*: BLOB type *BINARY*: BINARY type *ADT*: ADT type <i>bb...bb</i> : Data code [*]
"because multi column was unsupported, ' <i>aa...aa</i> ' was printed instead of the value."	There is a repetition column. <i>aa...aa</i> : *MCOL*

#

The data code indicates the following column data type:

- 0x65: INTERVAL YEAR TO DAY
- 0x6F: INTERVAL HOUR TO SECOND
- 0x71: DATE
- 0x79: TIME
- 0x7d: TIMESTAMP
- 0x83: ADT
- 0x91: BINARY
- 0x93: BLOB
- 0xE5: DECIMAL

S: Resumes processing.

KFRB02602-E

Error occurred in SQL output function, info = "*aa...aa*". (E)

An error was detected during update-SQL output processing.

aa...aa: Detail information

Message	Description
<code>fopen error, file = aa...aa, errno = bb...bb.</code>	File manipulation failed. Eliminate the cause of the error on the basis of the error number. For details about the error number, see <code>error.h</code> or the applicable OS documentation. <i>aa...aa</i> : File name [#] <i>bb...bb</i> : Error number
<code>fprintf error, file = aa...aa, errno = bb...bb.</code>	
<code>fflush error, file = aa...aa, errno = bb...bb.</code>	
<code>sprintf error, errno = bb...bb.</code>	
<code>internal error, file = aa...aa, line = bb...bb, detail = cc...cc.</code>	Contact the customer support center. <i>aa...aa</i> : Source file name <i>bb...bb</i> : Line number <i>cc...cc</i> : Internal code

#

This is a relative path from the source Datareplicator directory.

S: Cancels processing.

O: Correct the error on the basis of the detail information, and then restart the transmission target.

KFRB02603-I

SQL statement is printed.(transaction(*aa...aa*, *bb...bb*) :
insert(*aa...aa*, *bb...bb*) : update(*aa...aa*, *bb...bb*) : delete(*aa...aa*, *bb...bb*)
: purge(*aa...aa*, *bb...bb*)) (E)

SQL statement was output.

The number of output SQL statements is displayed as follows:

aa...aa: Number of statements issued at this time

bb...bb: Cumulative number of statements that have been output

S: Resumes processing.

KFRB02604-I

Event statement is printed, event code = *aa...aa*. (E)

An event statement was output.

aa...aa: Event code

S: Resumes processing.

KFRB02605-W

Reduction status of <send identifier: *aa...aa*> is continued. (E)

The reduced mode of the transmission target identifier *aa...aa* continues.

aa...aa: Transmission target identifier

S: Cancels update-SQL output processing for the corresponding transmission target.

KFRB03001-I

Reflect process initialization started. (C)

Import process initialization processing has started.

S: Resumes processing.

KFRB03002-I

Reflect process is now starting database update, mode = [nn...nnnxx...xx, m]. (C)

The import process will start updating the target database.

nn...nn: Start mode type 1:

Initial: Initial start

Normal: Normal start

Recover: Rerun start

xx...xx: Whether the import suppression function is used

Skip: The import suppression function is used.

Blank: The import suppression function is not used.

m: Start mode type 2:

N: Start from normal status

C: Start from critical status

S: Resumes processing.

KFRB03004-W

INSERT operation might occur DB contradiction, table name = aa...aa. (C)

The same row might have been imported twice by the INSERT operation during restart in the critical status.

aa...aa: Table name

S: Resumes processing.

[Action]

See the unimported information file to determine whether the table contents are correct.

KFRB03005-I

Reflect process ended critical session. Changing normal session. (C)

The import process's critical session has been completed.

The import process terminated import processing in the critical status. Import processing will now change to normal status.

S: Resumes processing.

KFRB03006-I

Reflect process caught event, event id = *nn...nn*. (C)

The import process detected an event.

nn...nn: Event code

Event code -1 indicates that an activity unit termination request from the source system was detected.

S: Resumes processing.

KFRB03007-W

Reflect process found automatically rollback transaction. Retrying the transaction. (C)

The import process detected a transaction that was rolled back implicitly. Datareplicator will re-execute the transaction.

S: Resumes processing.

[Action]

Contention has occurred between locked resources. Check the referencing program and change a lock search to a no-lock search, if possible.

KFRB03008-I

Reflect process reached the end point of received data, file information = [*mm...mm*, *nn...nn*]. (C)

The import process finished reading all the received update information.

mm...mm: File ID

nn...nn: File offset

S: Resumes processing. Datareplicator goes into monitoring status until it receives the next data.

KFRB03009-I

Reflect process completed commitment, reason = *aa...aa*, file information = [*mm...mm*, *nn...nn*]. (C)

The import process has completed synchronization point processing.

Detail information:

aa...aa: Reason for executing synchronization point processing:

COMMIT INTERVAL: Specified `cmtintvl` value was reached.
DISCONNECT INTERVAL: Specified `discintvl` value was reached.
MAX SQL NUMBER: Specified `reflect_trn_max_sqlnum` value was reached.
EVENT INFORMATION: Event was detected.
END OF CONNECTION: End of one transmission with the source system was detected.
END POINT OF QUEUE: End of update information was detected.
PURGE TABLE REQUEST: PURGE TABLE was detected.[#]
AFTER PURGE TABLE: PURGE TABLE was detected (after execution of PURGE TABLE)
DEFINITION REQUEST: Change in a definition was detected.
COMMAND REQUEST: Command termination was detected.
SYNC EVENT: A synchronous event was detected.

#

The number of updating operations displayed in this message does not include PURGE TABLE because the synchronization point processing applies only to the updating operations up to the PURGE TABLE that triggered the processing.

mm...mm: File ID*nn...nn*: File offset

Detail information:

Read transaction count = *xx...xx(yy...yy)*
SQL count = [Insert:*xx...xx(yy...yy)*, Update:*xx...xx(yy...yy)*,
Delete:*xx...xx(yy...yy)*, Purge: *xx...xx(yy...yy)*,
timestamp:*xx...xx(yy...yy)*, Commit: *xx...xx(yy...xx)*]
Additional Transaction Info = *zz...zz*.
Read transaction count: Number of transactions entered
xx...xx: Number of transactions entered since the previous synchronization point processing
yy...yy: Number of transactions entered since the import process started
SQL count: Number of SQL statements issued

Insert: Number of Insert statements issued.
Update: Number of Update statements issued
Delete: Number of Delete statements issued
Purge: Number of Purge statements issued
timestamp: Number of Insert statements issued to time-ordered table
Commit: Number of Commit statements issued
xx...xx: Count since the previous synchronization point processing
yy...yy: Count since startup of the import process

Additional Transaction Info: Extraction transaction information (zz...zz)

S: Resumes processing. Datareplicator displays detail information indicating the number of SQL statements issued before synchronization point processing occurred.

KFRB03010-W

No row satisfying search condition in reflect table. Then the SQL statement skipped, table name = aa...aa, SQLKIND = bb...bb, database status = cc...cc. (C)

Datareplicator ignored this SQL statement because the target table contained no rows that satisfy the specified condition.

aa...aa: Table name

bb...bb: SQL type

cc...cc: Database status:

NORMAL: Normal status

CRITICAL: Critical status

S: Resumes processing.

[Action]

See the unimported information file to determine whether the table contents are correct.

KFRB03011-I

Reflect process is now terminating database update, mode = nm...nm. (C)

Import processing termination processing is starting.

nm...nm: Termination mode:

Normal: Normal termination

Immediate: Termination by an immediate termination request

Event: Termination by event detection

Force: Forced termination

Defserv: Termination by a definition detection

S: Resumes processing.

KFRB03012-I

Reflect process terminated. (C)

The import process has now terminated.

S: Terminates processing.

KFRB03013-I

Reflect process terminated abnormally. (C)

The import process terminated abnormally because an error was detected.

S: Terminates processing.

KFRB03014-I

Reflect process aborted due to unexpected extract data received, additional information = [n, nn...nn, nn...nn]. (C)

The import process terminated abnormally because an error was detected in the update information.

n, nn...nn, nn...nn: File offset (internal information)

S: Cancels processing.

O: Obtain necessary information, such as a core dump, and contact the customer support center.

KFRB03015-E

Unable to continue Reflect process due to abnormal termination of child process. (C)

The import process cannot continue because termination of its subprocess was detected.

S: Terminates processing.

O: Check the message that was output by the SQL or UOC execution process. Obtain a core dump, if available, and contact the customer support center.

KFRB03016-E

Can not change number of group on rerun mode. (C)

You cannot change the key group in the rerun mode.

S: Terminates processing.

[Action]

Restore the import definition to its status during the previous execution and re-execute import processing.

KFRB03017-I

Reflect process completed transaction recovery, kind = *aa...aa*, file information = [*bb...bb*, *cc...cc*]. (C)

The import process completed transaction recovery.

aa...aa: Recovery type:

COMMIT: Commit recovery

COMMIT & RESTART: Commit recovery combined with rollback recovery

ROLLBACK: Rollback recovery

bb...bb and *cc...cc*: Queue file information

S: Resumes processing. In the case of commit recovery, Datareplicator re-executes the next transaction; otherwise, Datareplicator re-executes the previous transaction.

KFRB03021-E

Bad extracted data received, data kind = *n*. (C)

An error was detected in the update information.

n: Type of update information (internal information)

S: Terminates processing (subsequently issues KFRB03014-I).

KFRB03022-W

Not found Table-id in reflect definition, Table-id = *nn...nn*. (C)

Datareplicator detected update information for a table that is not defined in the import definition.

nn...nn: Table ID (internal information)

(If the table-based import method is being used, the table ID is output from one of the import processes.)

S: Continues processing.

[Action]

If you specified `defmerge=false` in the import environment definition, you can ignore this message. If you specified `defmerge=true`, the extraction definition contradicts the update information. Terminate the target Datareplicator, synchronize the data linkage environments at the source and target, initialize them, and then re-create the target database on the basis of the source database.

KFRB03023-E

Row length between extracted data and extract definition unmatched, table name = *aa...aa*. (C)

Datareplicator detected update information whose row length does not match the extraction definition.

aa...aa: Table name

S: Starts termination processing.

[Action]

The extraction definition contradicts the update information. Terminate the target Datareplicator, synchronize the data linkage environments at the source and target, initialize them, and then re-create the target database on the basis of the source database. Additionally, check the source system for operational errors.

KFRB03024-E

Bad column type found in extract definition, table name = *aa...aa*.
(C)

Invalid column attribute was detected in the extraction definition.

aa...aa: Table name

S: Starts termination processing.

[Action]

The extraction definition contradicts the update information. Terminate the target Datareplicator, synchronize the data linkage environments at the source and target, initialize them, and then re-create the target database on the basis of the source database. Additionally, check the source system for operational errors.

KFRB03025-E

Could not convert to target database language, table name =
aa...aa. (C)

Datareplicator was unable to convert the update information to the target database's character code set.

aa...aa: Table name

S: Terminates processing.

[Action]

Re-execute import processing. If this message is reissued, contact the customer support center.

KFRB03026-E

Number of extracted column data is not equal to extract
definition, table name = *aa...aa*. (C)

Datareplicator detected update information that was missing column information required for INSERT.

aa...aa: Table name

S: Starts termination processing.

[Action]

The extraction definition contradicts the update information. Terminate the target Datareplicator, synchronize the data linkage environments at the source and target, initialize them, and then re-create the target database on the basis of the source database. Additionally, check the source system for operational errors.

KFRB03027-E

Multi-value/Array column data found, table name = *aa...aa*. (C)

Datareplicator detected update information that includes a repetition column or an array column.

aa...aa: Table name

S: Starts termination processing.

[Action]

You cannot import a repetition column (when the Datareplicator version is earlier than 05-02) or an array column. Terminate the target Datareplicator, synchronize the data linkage environments at the source and target, initialize them, and then re-create the target database on the basis of the source database.

Delete the repetition or array column from the extraction definition or create a UOC routine to modify the import definition.

KFRB03028-W

Data convert error occurred. Reflect process set null value and continue, extract data name = *aa...aa*, reflect table name = *bb...bb*, reflect column name = *cc...cc*. (C)

A data conversion error occurred. The import process resumes processing assuming the null value for the invalid extraction data.

aa...aa: Update information name

bb...bb: Table name subject to import processing

cc...cc: Column name subject to import processing

S: Resumes processing.

[Action]

See the unimported information file to identify the erroneous extraction data. If necessary, update the data manually.

KFRB03029-E

Data convert error occurred, extract data name = *aa...aa*, reflect table name = *bb...bb*, reflect column name = *cc...cc*. (C)

A data conversion error occurred.

aa...aa: Update information name

bb...bb: Table name subject to import processing

cc...cc: Column name subject to import processing

S: Cancels processing.

[Action]

Terminate the target Datareplicator, synchronize the data linkage environments at the source and target, initialize them, and then re-create the target database on the basis of the source database

See the unimported information file to identify the erroneous column and either delete it from the update information or use a UOC routine to import it.

KFRB03030-W

Could not convert to target database language. Reflect process skipped this extract data, table name = *aa...aa*, column name = *bb...bb*. (C)

Datareplicator was unable to convert update information to the target database's character code set. Datareplicator will ignore this update information and continue import processing.

aa...aa: Source table name

bb...bb: Source column name

Detail information: Displays as a hexadecimal character string the first 50 bytes of data that produced the character code conversion error.

S: Resumes processing.

O: See the detail information to determine the cause of the code conversion error. If necessary, re-create the affected target table on the basis of the corresponding source table.

KFRB03031-E

HiRDB data base CONNECT error occurred, authorization id = *aa...aa*, SQLCODE = *nn...nn*. (C)

An error occurred while executing HiRDB CONNECT processing.

HiRDB CONNECT resulted in an error.

aa...aa: Authorization identifier

nn...nn: SQL code

Detail information: Displays a HiRDB message indicating the cause of the error.

S: Cancels processing.

[Action]

Eliminate the cause of the error and re-execute import processing.

KFRB03032-E

HiRDB data base PREPARE error occurred, table name = *aa...aa*,
SQLCODE = *nn...nn*. (C)

HiRDB PREPARE processing resulted in an error.

aa...aa: Table name

nn...nn: SQL code

Detail information: Displays a HiRDB message indicating the mapping key information and the cause of the error.

S: Cancels processing.

[Action]

Eliminate the cause of the error and re-execute import processing.

KFRB03033-E

HiRDB data base EXECUTE error occurred, table name = *aa...aa*,
SQLCODE = *nn...nn*. (C)

HiRDB EXECUTE processing resulted in an error.

aa...aa: Table name

nn...nn: SQL code

Detail information: Displays a HiRDB message indicating the mapping key information and the cause of the error.

S: Cancels processing.

[Action]

Eliminate the cause of the error and re-execute import processing.

The following table shows example actions.

SQL code	Action
100, -803	<ol style="list-style-type: none"> 1. If KFRB03007-W was issued before this message, the ROLLBACK processing might still be underway on HiRDB. Terminate the target Datareplicator, and then restart it after no more transactions whose PROGRAM name begins with hdssqle are found by the pdls -d trn -a command. 2. There might be an inconsistency between the source and target databases. Locate the erroneous row from the contents of the unimported information file, check and correct the contents of the source and target databases, and then restart the import processing. 3. A unique index for the table subject to extraction might not match the mapping key specified in the extraction definition. Correct the specification so that the unique index for the table subject to extraction matches the mapping key specified in the extraction definition, and then initialize Datareplicator. For details about the initialization procedure, see 6.2 <i>Initialization procedure at the environment configuration stage</i>.
-912	<ol style="list-style-type: none"> 1. The cmtintvl operand value in the import environment definition might be too large. Specify a smaller value in the cmtintvl operand, and then restart the import processing. If the cmtintvl operand value is 1, increase the target database's locked resources, and then restart the import processing.

KFRB03034-E

HiRDB data base EXECUTE IMMEDIATE error occurred, table name = aa...aa, SQLCODE = nn...nn. (C)

HiRDB EXECUTE IMMEDIATE processing resulted in an error.

aa...aa: Table name

nn...nn: SQL code

Detail information: Displays a HiRDB message indicating the mapping key information and the cause of the error.

S: Cancels processing.

[Action]

Eliminate the cause of the error and re-execute import processing.

KFRB03035-E

HiRDB data base COMMIT error occurred, SQLCODE = nn...nn. (C)

HiRDB COMMIT processing resulted in an error.

nn...nn: SQL code

Detail information: Displays a HiRDB message indicating the cause of the error.

S: Cancels processing.

[Action]

Eliminate the cause of the error and re-execute import processing.

KFRB03036-E

HiRDB data base DISCONNECT error occurred, SQLCODE = *nn...nn*. (C)

HiRDB DISCONNECT processing resulted in an error.

nn...nn: SQL code

Detail information: Displays a HiRDB message indicating the cause of the error.

S: Cancels processing.

[Action]

Eliminate the cause of the error and re-execute import processing.

KFRB03037-E

HiRDB data base ROLLBACK error occurred, SQLCODE = *nn...nn*. (C)

HiRDB ROLLBACK processing resulted in an error.

nn...nn: SQL code

Detail information: Displays a HiRDB message indicating the cause of the error.

S: Cancels processing.

[Action]

Eliminate the cause of the error and re-execute import processing.

KFRB03038-W

Update data for reflect column not exist in update information, extracted name = *aa...aa*, reflect table name = *bb...bb*. (C)

Update data for the target column was not found in the update information.

aa...aa: Update information name

bb...bb: Table name subject to import processing

S: Continues processing without importing this update information because data is not updated in the target column.

[Action]

If necessary, check for an error in the extraction definition and import definition.

KFRB03039-E

HiRDB data base SQL error occurred, SQLKIND = *aa...aa*. (C)

SQL processing on HiRDB resulted in an error.

aa...aa: SQL type

"TRG": Trigger processing

"AUD": Audit trail processing

S: Cancels processing.

[Action]

Eliminate the cause of the error, and then re-execute the command.

When the SQL type is AUD:

Check the `hirdb_audit_trail` operand value in the extraction system definition or the import system definition. If you want to collect an audit trail, change the operand value.

If there is no problem with the `hirdb_audit_trail` operand value, check the HiRDB message, and then eliminate the cause of the error.

KFRB03040-W

Reflect user own coding function returned warning code, process name = *aa...aa*, status = *nn...nn*. (C)

The import information updating UOC routine returned a warning-level status.

aa...aa: Import UOC process name

nn...nn: Status code returned from the UOC routine

S: Resumes processing.

[Action]

Check the contents of the import information editing UOC routine and take appropriate action.

KFRB03041-E

Reflect user own coding function returned error code, process name = *aa...aa*, status = *nn...nn*. (C)

The import information updating UOC routine returned an error-level status.

aa...aa: Import UOC process name

nn...nn: Status code returned from the UOC routine

S: Resumes processing.

[Action]

Eliminate the cause of the error and re-execute import processing.

KFRB03042-W

HiRDB data base EXECUTE error occurred. Reflect process skip this SQL and continue, table name = *nn...nn*, SQLCODE = *aa...aa*. (C)

An error occurred during EXECUTE SQL processing, but Datareplicator skipped this SQL statement and resumed processing.

nn...nn: Target table name

aa...aa: SQL code

S: Resumes processing.

KFRB03043-W

HiRDB data base EXECUTE IMMEDIATE error occurred. Reflect process skip this SQL and continue, table name = *nn...nn*, SQLCODE = *aa...aa*. (C)

An error occurred during EXECUTE IMMEDIATE SQL processing, but Datareplicator skipped this SQL statement and resumed processing.

nn...nn: Target table name

aa...aa: SQL code

S: Resumes processing.

KFRB03044-E

Column data UOC function returned error code, UOC function name = *xx...xx*, reflection table name = *mm...mm*, reflection column name = *nn...nn*, status = *aa...aa*. (C)

The column data editing UOC function returned an error-level status.

xx...xx: Name of the column data editing UOC function

mm...mm: Target table name

nn...nn: Target column name

aa...aa: Status returned from the column data editing UOC routine

S: Cancels processing.

[Action]

Eliminate the cause of the error and re-execute import processing.

KFRB03045-E

Invalid column data type set in column data UOC function, UOC function name = *xx...xx*, reflection table name = *mm...mm*, reflection column name = *nn...nn*. (C)

The column data editing UOC function specified an invalid column data type.

xx...xx: Name of the column data editing UOC function

mm...mm: Target table name

nn...nn: Target column name

S: Cancels processing.

[Action]

Eliminate the cause of the error and re-execute import processing.

KFRB03046-E

Invalid column data length set in column data UOC function, UOC function name = *xx...xx*, reflection table name = *mm...mm*, reflection column name = *nn...nn*. (C)

The column data editing UOC function specified an invalid length of the data.

xx...xx: Name of the column data editing UOC function

mm...mm: Target table name

nn...nn: Target column name

S: Cancels processing.

[Action]

Eliminate the cause of the error and re-execute import processing.

KFRB03047-W

HiRDB data base PREPARE error occurred. Reflect process skip this SQL and continue, table name = *nn...nn*, SQLCODE = *aa...aa*. (C)

An error occurred during PREPARE SQL processing, but Datareplicator skipped this SQL statement and resumed processing.

nn...nn: Target table name

aa...aa: SQL code

S: Resumes processing.

KFRB03048-E

HiRDB data base set 'W' in SQLWARN7, table name = *nn...nn*. (C)

W was set in SQLWARN7 during SQL execution.

Detail information: Mapping key information

Datareplicator ignored import processing on a repetition column because the element specified for the repetition column was not found in the corresponding row.

nn...nn: Target table name

S: Cancels processing.

[Action]

Eliminate the cause of the error and re-execute import processing.

KFRB03049-W

HiRDB data base set 'W' in SQLWARN7. Reflect process skipped this SQL and continue, table name = *nn...nn*. (C)

W was set in SQLWARN7 during SQL execution, but Datareplicator skipped this SQL statement and resumed processing.

Detail information: Mapping key information

Import processing on a repetition column was ignored because the element specified for the repetition column was not found in the corresponding row. Datareplicator skips this SQL statement and continues processing.

nn...nn: Target table name

S: Resumes processing.

KFRB03051-E

Data base CONNECT error occurred, data base = *aa...aa*, authorization id = *nn...nn*, error code = *bb...bb*. (C)

An error occurred during DBMS CONNECT processing.

aa...aa: DBMS type

nn...nn: Authorization identifier

bb...bb: DBMS error code

S: Cancels processing.

[Action]

See the DBMS manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03052-E

Data base PREPARE error occurred, data base = *aa...aa*, table name = *nn...nn*, error code = *bb...bb*. (C)

An SQL statement analysis error occurred in a DBMS.

aa...aa: DBMS type

nn...nn: *authorization-identifier.table-name*

bb...bb: DBMS error code

S: Cancels processing.

[Action]

See the applicable DBMS manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03053-E

Data base EXECUTE error occurred, data base = *aa...aa*, table name = *nn...nn*, error code = *bb...bb*. (C)

An SQL execution error occurred in a DBMS.

aa...aa: DBMS type

nn...nn: *authorization-identifier.table-name*

bb...bb: DBMS error code

S: Cancels processing.

[Action]

See the applicable DBMS manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03054-E

Data base ROLLBACK error occurred, data base = *aa...aa*, error code = *bb...bb*. (C)

An error occurred during ROLLBACK processing on a DBMS.

aa...aa: DBMS type

nn...nn: Authorization identifier

bb...bb: DBMS error code

S: Cancels processing.

[Action]

See the applicable DBMS manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03055-E

Data base COMMIT error occurred, data base = *aa...aa*, error code = *bb...bb*. (C)

An error occurred during COMMIT processing on a DBMS.

aa...aa: DBMS type

bb...bb: DBMS error code

S: Cancels processing.

[Action]

See the applicable DBMS manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03056-E

Data base DISCONNECT error occurred, data base = *aa...aa*, error code = *bb...bb*. (C)

An error occurred during DISCONNECT processing on a DBMS.

aa...aa: DBMS type

bb...bb: DBMS error code

S: Cancels processing.

[Action]

See the applicable DBMS manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03057-W

Data base EXECUTE error occurred, Reflect process skip this SQL and continue, data base = *aa...aa*, table name = *bb...bb*, error code = *cc...cc*. (C)

An error occurred during SQL processing on a DBMS, but Datareplicator skipped this SQL statement and resumed processing.

aa...aa: DBMS type

bb...bb: Table name

cc...cc: DBMS error code

S: Resumes processing.

KFRB03058-W

Data base CONNECT error occurred. Retry connection to database, data base = *aa...aa*, authorization id = *bb...bb*, error code = *cc...cc*. (C)

An error occurred during CONNECT processing on a DBMS. Datareplicator will re-execute CONNECT processing.

aa...aa: DBMS type

bb...bb: Authorization identifier

cc...cc: DBMS error code

S: Resumes processing.

KFRB03060-W

Reflect process terminated because HiRDB data base set 'W' in SQLWARN6, table name = *nn...nn*, SQLCODE = *aa...aa*. (C)

An SQL error occurred on the SQLCODE that is to be skipped, but processing was terminated because SQLWARN6 (implicit rollback) occurred.

nn...nn: Target table name

aa...aa: SQL code

S: Cancels processing.

O: Eliminate the cause of the SQL error, and then re-execute import processing.

KFRB03070-E

Error occurred in XA mode start. (C)

An error occurred during start in the XA mode.

S: Cancels processing.

O: Check the target HiRDB's message, eliminate the cause of the error, and then re-execute.

KFRB03071-E

Commitment prepare error occurred in transaction, XID = *aa...aa*. (C)

An error occurred during transaction commit preparation.

aa...aa: Transaction identifier

S: Cancels processing.

O: Check the target HiRDB's message, eliminate the cause of the error, and then re-execute.

KFRB03072-E

Commitment error occurred in transaction, XID = *aa...aa*. (C)

An error occurred during transaction commit processing.

aa...aa: Transaction identifier

S: Cancels processing.

O: Check the target HiRDB's message, eliminate the cause of the error, and then re-execute.

KFRB03073-E

Rollback error occurred in transaction, XID = *aa...aa*. (C)

An error occurred during transaction rollback processing.

aa...aa: Transaction identifier

S: Cancels processing.

O: Check the target HiRDB's message, eliminate the cause of the error, and then re-execute.

KFRB03074-E

Error occurred in HiRDB interface function, function = *aa...aa*, value = *bb...bb*. (C)

The HiRDB interface function resulted in an error.

aa...aa: Function name

bb...bb: Function's return value

S: Cancels processing.

O: Contact the customer support center.

KFRB03075-E

HiRDB don't support 2 phase commitment. (C)

The target HiRDB does not support the two-phase commit method.

fxa_sqle or *fxa_all* is specified for *commit_method* in the import system definition, but the target HiRDB does not support the two-phase commit method. The *PATH* or *SHLIB_PATH* environment variable might not have been specified correctly; or, library load processing might have failed due to a memory shortage.

S: Cancels processing.

O: Check the HiRDB version, the specification of *commit_method*, and the specification of the *PATH* or *SHLIB_PATH* environment variable, and then re-execute.

KFRB03076-W

HiRDB cannot accept the recovery request of the transaction. Retrying the recovery request, XID = *aa...aa*, function name = *bb...bb*, return = *cc...cc*. (C)

HiRDB could not accept the transaction recovery request. The recovery request will be reissued.

aa...aa: Transaction identifier

bb...bb: Function name

cc...cc: Function's return value

S: Resumes processing.

KFRB03077-I

HiRDB accepted the recovery request of the transaction, XID = *aa...aa*, function name = *bb...bb*, return = *cc...cc*. (C)

HiRDB accepted the transaction recovery request.

aa...aa: Transaction identifier

bb...bb: Function name

cc...cc: Function's return value

S: Resumes processing.

KFRB03078-E

In spite of reach maximum retry times, HiRDB cannot accept the recovery request of the transaction, XID = *aa...aa*, function name = *bb...bb*, return = *cc...cc*, retry count = *dd...dd*. (C)

The maximum retry count was reached, but HiRDB could not accept the transaction recovery request.

aa...aa: Transaction identifier

bb...bb: Function name

cc...cc: Function's return value

dd...dd: Retry count

S: Cancels processing.

[Action] Increase the `xa_recovery_retry_count` and `xa_recovery_retry_interval` operand values in the import environment definition. Once the target HiRDB has started, restart the import processing.

KFRB03080-E

Environment handle allocation error occurred. (C)

An error occurred while setting up the ODBC environment handle.

S: Cancels processing.

[Action]

See the ODBC manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03081-E

Connection handle allocation error occurred. SQLSTATE = *aa...aa*. (C)

An error occurred while setting up the ODBC connection handle.

aa...aa: SQLSTATE

S: Cancels processing.

[Action]

See the ODBC manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03082-E

Statement handle allocation error occurred. SQLSTATE = *aa...aa*. (C)

An error occurred while setting up the ODBC statement handle.

aa...aa: SQLSTATE

S: Cancels processing.

[Action]

See the ODBC manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03083-E

Unable to get information of ODBC driver, SQLSTATE = *aa...aa*. (C)
Datareplicator was unable to read the ODBC driver information.

aa...aa: SQLSTATE

S: Cancels processing.

[Action]

See the ODBC manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03084-E

Unable to set attributes of connections, Attribute = *aa...aa*, Value = *bb...bb*, SQLSTATE = *cc...cc*. (C)

Datareplicator was unable to set up the Connect attribute.

aa...aa: Attribute

bb...bb: Value

cc...cc: SQLSTATE

S: Cancels processing.

[Action]

See the ODBC manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03085-W

ODBC function returned warning code, function name = *aa...aa*, SQLSTATE = *bb...bb*. (C)

A warning occurred in an ODBC function.

aa...aa: ODBC function name

bb...bb: SQLSTATE

S: Resumes processing.

KFRB03086-E

Data base CONNECT error occurred, datasource = *aa...aa*, authorization id = *bb...bb*, SQLSTATE = *cc...cc*, error code = *dd...dd*. (C)

An error occurred during ODBC `CONNECT` processing.

aa...aa: Data source name

bb...bb: Authorization identifier

cc...cc: SQLSTATE

dd...dd: DBMS error code

S: Cancels processing.

[Action]

See the ODBC manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03087-E

Invalid ODBC version of ODBC driver, version = *aa...aa*. (C)

The ODBC driver's ODBC version is invalid.

aa...aa: ODBC version

S: Cancels processing.

[Action]

Check the supported versions of the ODBC driver that is being used.

KFRB03088-E

Invalid ODBC API conformance level of ODBC driver, conformance level = *aa...aa*. (C)

The ODBC driver's ODBC conformance level is invalid.

aa...aa: Conformance level

S: Cancels processing.

[Action]

Check your ODBC driver's conformance level.

KFRB03089-E

ODBC statement handle free error occurred, SQLSTATE = *aa...aa*, OPTION = *bb...bb*. (C)

An error occurred while freeing the ODBC statement handle.

aa...aa: SQLSTATE

bb...bb: SQLFreeStmt() option

S: Cancels processing.

[Action]

See the ODBC manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03090-E

ODBC function returned error code, function name = *aa...aa*,
SQLSTATE = *bb...bb*. (C)

An ODBC function resulted in an error.

aa...aa: ODBC function name

bb...bb: SQLSTATE

S: Cancels processing.

[Action]

See the ODBC manual and eliminate the cause of the error, and then re-execute import processing.

KFRB03091-W

Unable to set ODBC timeout, SQLSTATE = *aa...aa*. (C)

Datereplicator was unable to specify timeout for the ODBC function.

aa...aa: SQLSTATE

S: Resumes processing using the default setting (no timeout).

KFRB03092-W

Data base CONNECT error occurred, Retry connection to database,
datasource = *aa...aa*, authorization id = *bb...bb*, SQLSTATE = *cc...cc*,
error code = *dd...dd*. (C)

An error occurred during ODBC CONNECT processing. Datereplicator will retry CONNECT processing.

aa...aa: Data source name

bb...bb: Authorization identifier

cc...cc: SQLSTATE

dd...dd: DBMS error code

S: Resumes processing.

KFRB03094-W

HiRDB data base SQL error occurred, SQLKIND = *aa...aa*. (C)

Although an error occurred during SQL processing on HiRDB, the system ignores this error and resumes processing.

aa...aa: SQL type

"AUD": Audit trail processing

S: Resumes processing.

[Action]

Eliminate the cause of the error, and then re-execute the command.

When the SQL type is AUD:

- Check the `hirdb_audit_trail` operand value in the import system definition. If you want to collect an audit trail, change the operand value to a value other than `none_cont` or `none_stop`.
- If SQL statements are not issued to HiRDB from an import information editing UOC routine, change the `hirdb_audit_trail` operand value to a value other than `none_cont` or `none_stop` or do not link a HiRDB library when you create the executable file for the import information editing UOC routine.
- If SQL statements are issued to HiRDB from an import information editing UOC routine, make sure that no SQL statement that violates the creation rules is issued, and, if necessary, correct the import information editing UOC routine.

If none of the above problems applies, check the HiRDB messages and eliminate the cause of the error.

KFRB03101-W

Row not found at UPDATE request and now changing INSERT, extracted name = *aa...aa*, reflect table name = *bb...bb*. (C)

There is no row subject to UPDATE. UPDATE is being changed to INSERT.

aa...aa: Update information name

bb...bb: Table name subject to import processing

The mapping key value is output as additional information.

S: Resumes processing.

KFRB03102-W

Row already exists at INSERT request and now changing UPDATE, extracted name = *aa...aa*, reflect table name = *bb...bb*. (C)

The row subject to INSERT already exists. INSERT is being changed to UPDATE.

aa...aa: Update information name

bb...bb: Table name subject to import processing

The mapping key value is output as additional information.

S: Resumes processing.

KFRB03103-W

Change DELETE request to UPDATE (NULL value) because table merge option, extracted name = *aa...aa*, reflect table name = *bb...bb*. (C)

Datereplicator changed DELETE to UPDATE (null value) on the basis of the merge table option.

aa...aa: Update information name

bb...bb: Table name subject to import processing

The mapping key value is output as additional information.

S: Resumes processing.

KFRB03104-E

Conversion failed from INSERT to UPDATE statement, extracted name = *aa...aa*, reflect table name = *bb...bb*. (C)

Datereplicator was unable to change INSERT to UPDATE for the corresponding rows.

aa...aa: Update information name

bb...bb: Table name subject to import processing

The mapping key value is output as additional information. In addition, a simple dump is output under the HDSPATH directory.

S: Terminates processing.

O: Save the simple dump and contact the customer support center.

KFRB03105-E

Conversion failed from DELETE to UPDATE statement, extracted name = *aa...aa*, reflect table name = *bb...bb*. (C)

Datereplicator was unable to change DELETE to UPDATE for the corresponding rows.

aa...aa: Update information name

bb...bb: Table name subject to import processing

The mapping key value is output as additional information. Also, a simple dump is output under the HDSPATH directory.

S: Terminates processing.

O: Save the simple dump and contact the customer support center.

KFRB03106-W

Though the DELETE request was changed to UPDATE(NULL value) request, the UPDATE(NULL value) request was skipped because of no row satisfying search condition, extracted name = *aa...aa*, reflect table name = *bb...bb*. (C)

Because the sqlconvopt2 option was specified for the merge table, DELETE was

converted to UPDATE with a NULL value, and then the SQL statement was issued. However, this SQL statement was ignored because the target table contained no row satisfying the condition.

aa...aa: Update information name

bb...bb: Target table name

The mapping key value is displayed as additional information.

S: Resumes processing.

KFRB03110-W

Log data is ADT attribute data specified INSERT. Reflect process set null value and continue, extract data name = *aa...aa*, reflect table name = *bb...bb*, reflect column name = *cc...cc*. (C)

Datareplicator executed INSERT with the attribute of an abstract data type specified according to the options specified for the merge table and time-ordered information table. Datareplicator assumes the null value for any invalid extraction data and resumes import processing.

aa...aa: Update information name

bb...bb: Table name subject to import processing

cc...cc: Target column name

S: Terminates processing.

O: See the import error information file to determine the extraction data and check the data. If necessary, update the data manually.

KFRB03111-E

Not exist hash function library, library name = *aa...aa*. (C)

The hash function library was not found.

aa...aa: Library name

S: Terminates processing.

[Action]

Obtain necessary information, such as error files, and contact the customer support center.

KFRB03112-E

Failed to load hash function library, library name = *aa...aa*, errno = *bb...bb*. (C)

A load error occurred in the hash function library.

aa...aa: Library name

bb...bb: Error number

S: Terminates processing.

[Action]

Obtain necessary information, such as error files, and contact the customer support center.

KFRB03113-E

Failed to find hash function symbol, function name = *aa...aa*, errno = *bb...bb*. (C)

Acquisition of a hash function symbol failed.

aa...aa: Function name

bb...bb: Error number

S: Terminates processing.

[Action]

Obtain necessary information, such as error files, and contact the customer support center.

KFRB03114-E

Hash function error occurred, return code = *aa...aa*. (C)

The hash function terminated with an error.

aa...aa: Return code

S: Terminates processing.

[Action]

Obtain necessary information, such as error files, and contact the customer support center.

KFRB03115-E

Invalid hash value returned from hash function, RDAREA number = *aa...aa*, RDAREA index = *bb...bb*. (C)

The hash function returned an invalid RDAREA specification order.

aa...aa: Number of RDAREAs

bb...bb: RDAREA specification order

S: Terminates processing.

[Action]

Obtain necessary information, such as error files, and contact the customer support center.

KFRB03116-W

Null value set in hash key column, hash key column name = *aa...aa*.
(C)

The null value was set in the hash key column.

aa...aa: Name of the hash key column

S: Resumes processing without importing the corresponding update information.

KFRB03117-E

Failed to get hash key column data, hash key column name = *aa...aa*.
(C)

Acquisition of the hash key column value failed.

aa...aa: Name of the hash key column

S: Terminates processing.

[Action]

Obtain necessary information, such as error files, and contact the customer support center.

KFRB03201-I

Definition Server process started. mode = *nn...nn*. (C)

The import definition server process has started.

nn...nn: Type of start mode:

Initial: Initial start

Normal: Normal start

Recover: Rerun start

S: Resumes processing.

KFRB03202-I

Definition Server process terminated, status = *nn...nn*, addinfo = *xx...xx*. (C)

The import definition server process has terminated.

nn...nn: Termination status:

Initial: Initial status (starts with initialization at restart)

Normal: Normal termination status

Immediate: Immediate termination status

Event: Event termination status

Error: Error termination status (executes rerun at restart)

xx...xx: Detail information (internal information)

S: Terminates processing.

[Action]

If the termination status is `Initial` or `Error`, an error might have occurred. Check the error message issued before this message. If an error occurred, eliminate the cause of the error, and then re-execute import processing.

KFRB03203-E

Definition Server process terminated abnormally, module = *aa...aa*, line = *nn...nn*, addinfo = *xx...xx*. (C)

The import definition server process terminated abnormally due to an error.

aa...aa: Module name

nn...nn: Line number

xx...xx: Detail information (internal information)

S: Cancels processing.

O: Obtain necessary information, such as a core dump, contact the customer support center.

KFRB03204-I

Now, watching reflect process. (C)

Datareplicator is monitoring the import process.

S: Resumes processing.

KFRB03205-I

Found abnormal termination of Reflect process, group name = *aa...aa*. (C)

Datareplicator detected abnormal termination of the import process.

aa...aa: Group name

S: Resumes processing.

KFRB03206-E

Reflection-env file parsing error occurred. (C)

An error occurred while analyzing the import environment definition file.

S: Starts termination processing.

[Action]

Correct the operand(s) specified in the import environment definition file.

KFRB03207-E

Internal error occurred in definition server, factor code = *aa...aa*, addinfo = [*mm...mm*, *nn...nn*]. (C)

The import definition server process resulted in an internal error.

aa...aa: Cause code

mm...mm, *nn...nn*: Detail information (internal information)

S: Terminates processing.

[Action]

Obtain necessary information, such as the import error information file, and contact the customer support center.

KFRB03208-I

Defer start of reflection, deferred time = *aa...aa* minutes, reflect mode = *nn...nn*. (C)

The start of import processing will be delayed.

aa...aa: Delay time

If this value is 0, import processing will be delayed until the import processing restart request is issued.

nn...nn: Import processing method after delay start:

TBL: Table-based import method

TRN: Transaction-based import method

SPD: Undetermined (the method will be determined when an import processing restart request is issued because neither the `breaktime` nor the `breakmode` operand is specified)

S: Resumes processing.

KFRB03209-I

Initialization completed in reflection. (C)

Initialization of import processing has been completed.

S: Resumes processing.

KFRB03210-I

Accepted stop request in reflector, request kind = *aa...aa*. (C)

An import processing stop request was accepted.

aa...aa: Request type:

Normal: Because Datareplicator accepted the `hdsstop` command, it will terminate processing after completing import processing through the last

end-of-transmission message.

Event: Because Datareplicator accepted the `hdsstop -t event` command, it will terminate processing after importing the first event that occurs from now on.

Force: Because Datareplicator accepted the `hdsstop -t force` command, it will terminate processing after database conformity is achieved.

Spd: Because Datareplicator accepted the `hdsrftcl -m spd` command, it will terminate processing after importing the first event that occurs from now on.

S-Immediate: Because Datareplicator accepted the `hdsstop -t immediate` command, it will wait for termination of the import process.

R-Immediate: Because Datareplicator accepted the `hdsrftcl -d data-linkage-identifier -m immediate` command, it will wait for termination of the import process.

G-Immediate: Because Datareplicator accepted the `hdsrftcl -g synchronous-import-group-name -m immediate` command, it will wait for termination of the import process.

S: Resumes processing.

KFRB03211-I

All reflect process caught event, event-id = *aa...aa*. (C)

All import processes have reached the indicated event.

aa...aa: Event ID

S: Resumes processing.

KFRB03213-W

Reflection skip information, extract id = *aa...aa*. (C)

Importing of the specified update information was suppressed.

aa...aa: Identifier of update information whose importing was suppressed

S: Resumes processing.

KFRB03214-E

Reflection skip list file definition error, file name = *aa...aa*, line no = *bb...bb*, reason = *cc...cc*. (C)

A definition error was detected in the import suppression list file.

aa...aa: Name of the import suppression list file

bb...bb: Number of the line where the error occurred

cc...cc: Reason code:

11: Analysis error

12: Duplicate registration error

13: Detection of unsupported control code

S: Terminates processing.

O: Correct the import suppression list file.

KFRB03301-I

Synchronization control process started. (C)

The synchronization managing process has started.

S: Resumes processing.

KFRB03302-I

Synchronization control process ended. (C)

The synchronization managing process has terminated.

S: Resumes processing.

KFRB03303-E

Exceeded limit time or transaction count of processing, detail = *aa...aa*. (C)

The maximum time to wait until synchronization was exceeded, or the maximum number of transactions to wait for until synchronization was exceeded.

aa...aa: Reason the maximum value was exceeded:

time: Maximum time to wait until synchronization

transaction count: Maximum number of transactions to wait for until synchronization

S: Cancels processing.

O: Take the following action:

When *aa...aa* is time:

Take one of the following actions:

- Check files such as the error information file, eliminate the cause of prolonged processing time, and then restart the import processing.
- Increase the `syncwait_limit_time` operand value in the import system definition, and then restart the target Datareplicator.

When *aa...aa* is transaction count:

Increase the `syncwait_limit_tran_count` operand value in the import system definition, and then restart the target Datareplicator.

If the error cannot be corrected by these actions, disable the synchronous import

group (`hdsstart -c synchronous-import-group-name`) at the target Datareplicator, and then restart the Datareplicator.

KFRB03304-E

Unable to specified dsid of Synchronization control process, group name = *aa...aa*, dsid = *bb*, command = *cc...cc*. (C)

The data linkage identifier constituting a synchronous import group cannot be manipulated.

aa...aa: Synchronous import group name

bb: Data linkage identifier

cc...cc: Command name

S: Resumes processing. However, the system will not accept commands.

O: Specify the synchronous import group name, and then re-execute the command.

KFRB03311-E

Unable to continue Reflect process due to abnormal termination of Synchronized control process. (C)

Termination of a synchronization managing process was detected. The system will terminate all import processing that constitutes the synchronous import group.

S: Cancels processing.

O: Eliminate the cause of the error according to the message that was displayed immediately before this message, and then use the `hdsstart` or `hdsrftcl` command to restart the synchronous import group.

KFRB03312-E

Detected illegal extract data. (C)

Datareplicator detected update information that is not supported by the import transaction synchronization facility. The `pd_rpl_reflect_mode` operand value might have been changed at the source HiRDB.

S: Cancels processing.

O: Use the `hdsstop` command to terminate the target Datareplicator, disable the synchronous import group (`hdsstart -c synchronous-import-group-name`), and then restart the Datareplicator.

KFRB03314-W

Executed purge table, table name = *aa...a*. (C)

Part of the import transaction was committed by execution of Purge table. From now on, synchronization of import transactions cannot be guaranteed until a synchronous event is detected.

aa...a: Name of the extraction table on which Purge table was executed

A table name followed by 30 bytes of spaces is output.

S: Resumes processing.

O: From now on, synchronization of import transactions cannot be guaranteed until a synchronous event is detected. Execute a synchronous event immediately at the source system.

KFRB03315-E

UOC cannot be executed. (C)

The import transaction synchronization facility does not support an import information editing UOC.

S: Cancels processing.

O: Delete the specification of by 'uoc-name' from the load statement in the import definition.

KFRB03316-W

Detected illegal event code, event code = aa...a. (C)

The import transaction synchronization facility does not support any event codes other than ones specified in eventsync.

aa...a: Detected event code

S: Ignores the indicated event code and resumes processing.

KFRB03317-E

Detected illegal extract definition. (C)

The extraction definition cannot be changed while the import transaction synchronization facility is in use.

S: Cancels processing.

O: Use the `hdsstop` command to terminate the target Datareplicator, disable the synchronous import group (`hdsstart -c synchronous-import-group-name`), and then restart the Datareplicator.

KFRB04001-E

File access error was occurred, operation = nnnnn, file = aa...aa, errno = xx...xx. (S)

A file manipulation error was detected.

nnnnn: Manipulation type (system call name)

aa...aa: Filename

xx...xx: Error number

Error number 0 indicates as follows:

If the manipulation type is `open`: The file has not been initialized or an initialization error occurred.

If the manipulation type is `read`: The file contents are invalid.

If the manipulation type is `write`: There is not enough disk space.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04002-E

Necessary file not found, file = *aa...aa*. (S)

A required file is missing.

aa...aa: Filename

S: Cancels processing.

O: See the following table and take appropriate action based on the command that was executed and the file name displayed in *aa...aa*:

Command	File name	Description	Action
<code>hdestart_ n</code>	Extraction system definition file name	The extraction system definition file was not found. Possible causes include: 1. An invalid directory was specified in the <code>HDEPATH</code> environment variable. 2. An attempt was made to start the extraction master process at a node that does not contain the system manager (the command was executed with the <code>-b</code> option omitted).	<ul style="list-style-type: none"> When the cause is 1 Specify the correct directory in the <code>HDEPATH</code> environment variable. When the cause is 2 To start the extraction master process, execute the command at the node where the system manager is located.

Command	File name	Description	Action
hdestop_n	errfile_1server-name	<p>The extraction node master error information file was not found. Possible causes include:</p> <ol style="list-style-type: none"> 1. An invalid directory was specified in the HDEPATH environment variable. 2. The value of the nodecontrol operand in the extraction system definition is unit. 3. The server specified in the -b option is not located at the node where the command was executed, or an invalid server name was specified in the -b option. 4. The extraction node master error information file was deleted illegally. 	<ul style="list-style-type: none"> • When the cause is 1 Specify the correct directory in the HDEPATH environment variable. • When the cause is 2 To terminate the source Datareplicator, use the hdestop command. For details about using the hdestop_n command, such as prerequisites, see 7. <i>Command Syntax</i>. • When the cause is 3 Check whether the command was executed at the node where the server specified in the -b option is located. • When the cause is 4 The hdestop_n command cannot be executed. Use the hdestop command to terminate the source Datareplicator.
	Extraction system definition file name	<p>The extraction system definition file was not found. Possible causes include:</p> <ol style="list-style-type: none"> 1. An invalid directory was specified in the HDEPATH environment variable. 2. An attempt was made to terminate the extraction master process at a node that does not contain the system manager (the command was executed with the -b option omitted). 	<ul style="list-style-type: none"> • When the cause is 1 Specify the correct directory in the HDEPATH environment variable. • When the cause is 2 Execute the command at the node where the system manager is located.
Other	--	A file required for command execution is missing.	Eliminate the cause of the error, and then re-execute the command.

KFRB04003-E

Invalid file size, file = *nm...nm*, size = *aa...aa*. (S)

A file size is invalid.

nm...nm: Filename

aa...aa: File size

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04004-E

Invalid file type, file = *aa...aa*. (S)

A type of file is invalid.

aa...aa: Filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04005-E

Invalid file data, file = *aa...aa*. (S)

The contents of a file are invalid.

aa...aa: Filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04006-E

Unable to use both UNIX regular file and Character special file.
(S)

You cannot mix UNIX regular files and character special files.

S: Cancels processing.

O: Correct the type of files to be used by referencing *4.6.2 Preparation of the files used with the source Datareplicator* or *4.7.2 Preparation of the files used with the target Datareplicator*, and then re-execute.

KFRB04007-E

Necessary directory not found, directory = *aa...aa*. (S+L)

A required directory is missing.

aa...aa: Directory name

S: Cancels processing.

O: See the following table and take appropriate action based on the command that was executed and the directory name displayed in *aa...aa*:

Command	Directory name	Description	Action
<code>hdestart_n</code>	Directory name specified in the <code>HDEPATH</code> environment variable	The source Datareplicator's directory was not found.	Specify the correct directory in the <code>HDEPATH</code> environment variable.

Command	Directory name	Description	Action
hdestop_n			

KFRB04011-E

Semaphore operation failure, sem-id = *aa...aa*, kind = *nm...nm*, errno = *xx...xx*. (S)

Lock processing failed.

aa...aa: Semaphore ID

nm...nm: Lock type

xx...xx: Error number set in errno

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04012-E

Unable to allocate semaphore, sem-key = *aa...aa*, size = *bb...bb*, errno = *cc...cc*. (S)

Semaphore allocation failed.

aa...aa: Semaphore key

bb...bb: Semaphore allocation size

cc...cc: Error number set in errno

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04021-E

Signal operation failure, process-id = *aa...aa*, kind = *bb...bb*, errno = *xx...xx*. (S)

Signal manipulation failed.

aa...aa: Remote process ID

bb...bb: Signal to be sent

xx...xx: Error number set in errno

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. If Datareplicator was started from JP1/Cm2, you must be a superuser to re-execute this processing.

If *errno=1* is displayed, the user who started the target process must re-execute the command.

KFRB04031-E

Cannot exec child process, process name = *aa...aa*, errno = *xx...xx*.
(C)

Datereplicator is unable to execute a subprocess.

aa...aa: Subprocess name

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04032-E

Cannot fork child process, errno = *xx...xx*. (S)

Datereplicator is unable to generate a subprocess.

xx...xx: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04033-E

Timeout occurred in Source site Datereplicator start, waiting
time = *aa...aa*second, process name = *bb...bb*. (S)

The source Datereplicator did not complete its start processing within *aa...aa* seconds.

aa...aa: Time to wait for completion of start processing

bb...bb: Name of the process to be started

S: Stops processing.

O: Contact the customer support center and obtain a core dump that has been output
under the directory used to execute the `hdestart` command.

KFRB04034-E

Found abnormal termination process, pid = *aa...aa*, process name =
bb...bb. (S)

Abnormal termination of a process was detected.

aa...aa: Process ID of the child process that terminated abnormally

bb...bb: Name of the process to be started

S: Stops processing.

O: Check for messages output to `syslogfile`, `msterrfile1`, or `msterrfile2`,
eliminate the cause of the error, and then re-execute the `hdestart` command. If no
message was output, contact the customer support center.

KFRB04041-E

Shared memory operation failure, type = *mm...mm*, shm-id = *aa...aa*, kind = *nn...nn*, errno = *xx...xx*. (S)

Manipulation of shared memory failed.

mm...mm: Type of shared memory

aa...aa: Shared memory ID

nn...nn: Manipulation type

xx...xx: Error number set in errno

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04042-E

Unable to allocate shared memory, type = *mm...mm*, shm-key = *aa...aa*, size = *bb...bb*, errno = *cc...cc*. (S)

Allocation of shared memory failed.

mm...mm: Type of shared memory

aa...aa: Shared memory key

bb...bb: Allocation size of shared memory

cc...cc: Error number set in errno

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04051-E

Insufficient memory, type = *mm...mm*, size = *nn...nn*, errno = *xx...xx*. (S)

There is not enough memory.

mm...mm: Detail code

nn...nn: Allocation size of memory

xx...xx: Error number set in errno

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04061-E

Necessary environment variable value cannot get, variable name = *aa...aa*. (S)

An environment variable required for manipulation is not specified.

aa...aa: Environment variable name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04062-E

Environment variable value invalid, variable name = *aa...aa*. (S)

The value specified for an environment variable is invalid.

aa...aa: Environment variable name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04071-E

Standard input not terminal. (S)

The standard input is not a terminal.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04101-E

Invalid command argument. (S)

Specified command argument is invalid.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04102-E

Invalid option specified, option name = *aa...aa*. (S)

An invalid command option was detected.

aa...aa: Command option

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04103-E

Necessary option not specified, option name = *aa...aa*. (S)

A required command option is missing.

aa...aa: Command option

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04104-E

Option value is invalid, option name = *aa...aa*. (S)

Value of a command option is invalid.

aa...aa: Command option

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04105-E

Options can not specify simultaneously, option name = *aa...aa* and *bb...bb*. (S)

Mutually exclusive command options are specified.

aa...aa and *bb...bb*: Command options

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04201-E

Invalid User-id found. (S)

User ID is invalid.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04202-E

Invalid Password found. (S)

Password is invalid.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04203-E

Invalid Time range in definition file, definition file = *aa...aa*. (S)

A time specified in a definition file is invalid.

aa...aa: Name of the import system definition file or import environment definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04204-E

Invalid HiRDB Datareplicator-id found in definition file, definition file = *aa...aa*. (S)

A target Datareplicator identifier specified in a definition file is invalid.

aa...aa: Name of the import system definition file or import environment definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04205-E

Invalid replication node-id found in definition file, definition file = *aa...aa*. (S)

A data linkage identifier specified in a definition file is invalid.

aa...aa: Name of the import system definition file or import environment definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04206-E

Invalid service name found in definition file, definition file = *aa...aa*. (S)

A service name specified in a definition file is invalid.

aa...aa: Name of the import system definition file or import environment definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04207-E

Invalid file name found in definition file, file name = *aa...aa*, definition file = *bb...bb*. (S)

A filename specified in a definition file is invalid.

aa...aa: Specified filename

bb...bb: Name of the import system definition file or import environment definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04208-E

Invalid T-SELECTOR found in definition file, definition file = *aa...aa*. (S)

T-selector specified in a definition file is invalid.

aa...aa: Name of the import system definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04211-E

Sector size is larger than queuesize, sector size = *aa...aa*,
 queuesize = *bb...bb*. (S)

The sector length is greater than the size of the import information queue file.

aa...aa: Sector length

bb...bb: Queue file size

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04212-E

Sector size is larger than statssize, sector size = *aa...aa*,
 statssize = *bb...bb*. (S)

The sector length is greater than the size of the import status file.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04213-E

Command option value is not equal to dsid number of incomplete
 initialization, previous value = *aa...aa*, current value = *bb...bb*.
 (S)

The specified data linkage identifier number is different from the one specified during
 the partial initialization that resulted in being incomplete.

aa...aa: Previous data linkage identifier number specified

bb...bb: Data linkage identifier number specified this time

S: Cancels processing.

O: Specify the previous data linkage identifier number and re-execute partial
 initialization; or, initialize the entire target Datareplicator.

KFRB04214-E

The initializing which designated dsid was incomplete, dsid no=
aa...aa. (S)

Partial initialization cannot be completed.

aa...aa: Data linkage identifier number specified during the previous partial
 initialization

S: Cancels processing.

O: Specify the previous data linkage identifier number and re-execute partial
 initialization; or, initialize the entire target Datareplicator.

KFRB04215-E

Target site Datareplicator is not initialized. (S)

The target Datareplicator has not been initialized.

S: Cancels processing.

O: Initialize the entire target Datareplicator.

KFRB04216-I

Target site Datareplicator initialization completed. (C)

Initialization of the target Datareplicator has been completed.

S: Terminates the system.

KFRB04217-E

Unable to initialize target site Datareplicator, because there are data to be reflected. (S)

The import environment cannot be initialized.

S: Cancels processing.

O: Take appropriate action according to the target Datareplicator's status.

Target Datareplicator's status	Action
<ul style="list-style-type: none"> There is still unimported update information that needs to be imported. 	Import all unimported update information, and then reinitialize Datareplicator.
<ul style="list-style-type: none"> There is still unimported update information, but there is no need to import it. All update information has been imported, but Datareplicator was terminated forcibly by the <code>hdsstop -t force</code> command. The import information status file or import information queue file was created as a character special file and Datareplicator is to be initialized for the first time. The status file name specified in the <code>statsfile</code> operand in the import environment definition is to be changed, but a file with the new status file name already exists. 	Reinitialize Datareplicator with the <code>hdsstart -i -f</code> command.

KFRB04218-I

Command request was ignored because process is running, process = `aa...aa`, pid = `bb...bb`. (S+L)

The start request was ignored because the specified process was already running.

`aa...aa`: Process type:

`hdemaster`: Extraction master process

`hdenodemst`: Extraction node master process

bb...bb: Process ID

S: Cancels processing.

KFRB04219-E

Command request was ignored because process is not running,
process = *aa...aa*. (S+L)

The termination request was ignored because the specified process was not running.

aa...aa: Process type:

hdemaster: Extraction master process

hdenodemst: Extraction node master process

Note:

If the `hdestop_n` command is executed in an environment in which all the conditions listed below are satisfied, this message is displayed as the communication execution result even when the extraction node master process is running:

- The `nodecontrol` operand value in the extraction system definition is `unit`.
- There is a file named `errfile1_server-name` under the operation directory.
- The `hdestop_n` com is executed with the `-b` operand specified.

Use the `hdestop` command to terminate the source Datareplicator in such a case.

For details about using the `hdestop_n` command, such as prerequisites, see 7. *Command Syntax*.

S: Cancels processing.

KFRB04220-E

Unable to execute *aa...aa* command because of definition, info = "*bb...bb*". (S+L)

The command cannot be executed because the Datareplicator definition does not satisfy the prerequisites for command execution.

aa...aa: Command name

bb...bb: Operand information

S: Cancels processing.

O: See the following table and take appropriate action based on the command and operand information:

Command	Directory name	Description	Action
hdestop_n	nodecontrol=unit	The command cannot be executed because the nodecontrol operand value in the extraction system definition is unit.	Use the hdestop command to terminate the source Datareplicator. For details about using the hdestop_n command, such as prerequisites, see 7. <i>Command Syntax</i> .

KFRB04221-E

Command request acceptance error was occurred, command = *aa...aa*, reason = *bb...bb*. (S)

A command request acceptance error occurred.

aa...aa: Command name

bb...bb: Reason code

S: Cancels processing.

O: See the following table and take appropriate action based on the command and the reason code:

Command	Reason code	Description	Action
hdsrftcl	1	A command argument is invalid	Check and, if necessary, revise the command argument. Take appropriate action based on the message that is output to the import error information file, syslog file, or standard error output.
	11	This command was executed with the import processing start option specified, but import processing was already underway.	There is no need to take any action because import processing has already started.
	12	This command was executed with the import processing start option specified, but there was no response from the import master process.	Check the import error information file to determine whether the target Datareplicator is running correctly. If no error message has been output, the target Datareplicator might be in a status that prevents it from sending a response. Execute the resource deletion command, and then restart the target Datareplicator.

Command	Reason code	Description	Action
hdechgststatus	11	Status updating failed on some nodes or servers.	Take the following action: <ol style="list-style-type: none"> 1. Check the KFRB00724-W message in the extraction master error information file to identify the name of the host that resulted in the update error. 2. If an error message was output immediately before the KFRB00724-W message, take appropriate action based on that error message. 3. If no error message has been displayed, check the KFRB00725-W message that was output to the extraction node master error information file to identify the name of the server that resulted in the update failure. Take appropriate action based on the error message that was displayed immediately before the KFRB00725-W message.

KFRB04301-E

Target site Datareplicator is already started. (S)

The target Datareplicator has already started.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04302-E

Target site Datareplicator is not started. (S)

The target Datareplicator has not been started.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04303-E

Same type command was already accepted. This command was ignored, accepted command = aa...aa. (S)

The same type of command is already executing. Datareplicator will ignore this command.

aa...aa: Command already accepted

S: Cancels processing.

KFRB04304-E

After HiRDB Datareplicator initialized, operand changed in definition file, operand = *aa...aa*, definition file = *bb...bb*. (S)

An operand in a definition file was changed after Datareplicator was initialized.

aa...aa: Operand name

If the displayed operand name is *refenv001* to *refenv128*, the *statsfile* operand in the import processing environment file might have been changed.

bb...bb: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04305-E

Update queue file must be more than two in definition file, definition file = *aa...aa*. (S)

You must provide at least two import information queue files.

aa...aa: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04306-E

Replication node-id not specified in definition file, definition file = *aa...aa*. (S)

No data linkage identifier is specified.

aa...aa: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04307-E

Protocol not specified in definition file, definition file = *aa...aa*. (S)

No protocol is specified.

aa...aa: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04308-E

Unreflected data file must be two in definition file, definition file = *aa...aa*. (S)

You must provide at least two unimported information files.

aa...aa: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04309-E

Reflect status file not specified in definition file, definition file = *aa...aa*. (S)

No import status file is specified.

aa...aa: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04310-E

Reflect environment file not specified for replication node-id in definition file, definition file = *aa...aa*. (S)

No import environment definition file corresponding to the data linkage identifier is specified.

aa...aa: Name of the import environment definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04311-E

File size not specified in definition file, file kind = *aa...aa*, definition file = *bb...bb* (S)

File size is not specified.

aa...aa: File type

bb...bb: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04312-E

Shared memory size not specified in definition file, definition file = *aa...aa*. (S)

The size of shared memory for storing definition information is not specified.

aa...aa: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04313-E

Cannot accept command while processing command, rejected command = *aa...aa*. (S)

Datareplicator cannot accept the command because command processing is already underway.

aa...aa: Command name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04314-E

File name not unique in definition file, file name = *aa...aa*. (S)

A filename is not unique in a definition file.

aa...aa: Filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04315-E

Replication node-id not unique in definition file, definition file = *aa...aa*. (S)

A data linkage identifier is not unique in a definition file.

aa...aa: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04316-E

Event code not unique in definition file, definition file = *aa...aa*. (S)

An event code is not unique in a definition file.

aa...aa: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04317-E

T-SELECTOR not specified in definition file, definition file = *aa...aa*. (S)

No T-selector is specified.

aa...aa: Name of the import system definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04318-E

Invalid SQLCODE found in skip_sqlcode, definition file = *nn...nn*.
(S)

Invalid SQLCODE was specified in skip_sqlcode.

nn...nn: Definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04319-E

oracleusr not specified in definition file, definition file =
aa...aa. (S)

oracleusr is not specified in the import system definition file.

aa...aa: Name of the import system definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04320-E

odbcusr not specified in definition file, definition file =
aa...aa. (S)

odbcusr is not specified in the import system definition file.

aa...aa: Name of the import system definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04321-E

datasource not specified in definition file, definition file =
aa...aa. (S)

datasource is not specified in the import system definition file.

aa...aa: Name of the import system definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04322-E

Duplicate file kind appointed in definition file, operand =
aa...aa, definition file = *bb...bb*. (S)

A type of file to be allocated to the Datareplicator file system area is duplicated.

aa...aa: Operand name

bb...bb: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04323-E

Invalid file kind appointed in definition file, file kind = *aa...aa*, definition file = *bb...bb*. (S)

The type of file to be allocated to the Datareplicator file system area is invalid.

aa...aa: Type of file to be allocated

bb...bb: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04324-E

Unable to use file system, file system = *aa...aa*, error code = *bb...bb*. (S)

A Datareplicator file system area is not usable.

aa...aa: Name of the Datareplicator file system area

bb...bb: Detail code for the Datareplicator file system area access error

S: Cancels processing.

O: Check that the file specified for the Datareplicator file system area has already been initialized, and then re-execute. For details about the detail codes for Datareplicator file system area access errors, see *KFRB00714-E*.

KFRB04325-E

Failed to initialize file system, file system = *aa...aa*, error code = *bb...bb*. (S)

A Datareplicator file system area is not usable.

aa...aa: Name of the Datareplicator file system area

bb...bb: Detail code for the Datareplicator file system area access error

S: Cancels processing.

O: Initialize the file specified for the Datareplicator file system area using the *hdsfmkfs* command, and then re-execute. For details about the detail codes for Datareplicator file system area access errors, see *KFRB00714-E*.

KFRB04326-E

Failed to prepare file system, file system = *aa...aa*, error code = *bb...bb*. (S)

A Datareplicator file system area is not usable.

aa...aa: Name of the Datareplicator file system area

bb...bb: Detail code for the Datareplicator file system area access error

S: Cancels processing.

O: Check that the file specified for the Datareplicator file system area is usable, and then re-execute. For details about the detail codes for Datareplicator file system area access errors, see *KFRB00714-E*.

KFRB04327-E

Unable to add file system, file system = *aa...aa*, add file = *bb...bb*, add size = *cc...cc*, error code = *dd...dd*. (S)

A Datareplicator file system area is not usable.

aa...aa: Name of the Datareplicator file system area

bb...bb: Name of the additional file

cc...cc: Size of the additional file

dd...dd: Detail code for the Datareplicator file system area access error

S: Cancels processing.

O: Check the size of the file to be stored in the Datareplicator file system area, and then re-execute. For details about the detail codes for Datareplicator file system area access errors, see *KFRB00714-E*.

KFRB04328-E

HiRDB don't support 2 phase commitment. (S)

fxa_sqle or *fxa_all* was specified for *commit_method* in the import system definition, but the target HiRDB does not support the two-phase commit method. The specified *PATH* or *SHLIB_PATH* environment variable might be invalid; or, a library load operation might have failed due to a memory shortage.

S: Cancels processing.

O: Check the HiRDB version, the specified *commit_method*, and the specified *PATH* and *SHLIB_PATH* environment variables, and then re-execute.

KFRB04329-E

Unable to share the file system with another *dsid*, file system = *aa...aa*, definition file = *bb...bb*. (S)

If partial initialization is executed, the Datareplicator file system area name cannot be shared with another *dsid*.

aa...aa: Name of the Datareplicator file system area

bb...bb: Definition filename

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04330-E

Insufficient status file size. (S)

The size of the import status file is inadequate.

S: Cancels processing.

O: Increase the *statssize* operand value in the import environment definition according to the formula for estimating the file size, and then re-execute the command.

Note that in Datareplicator 06-03 or later, the minimum size is 165 kilobytes.

KFRB04331-Q

Do you really want to clean up all resources (y or n)? (S)

Enter a response to whether Datareplicator's shared resources are to be deleted.

- If *y* is entered: Datareplicator deletes its shared resources.
- If *n* is entered: Datareplicator cancels the deletion processing on its shared resources, and then terminates the processing.

S: Waits for a response.

O: Enter *y* to delete Datareplicator's shared resources or *n* to cancel the deletion processing on Datareplicator's shared resources.

KFRB04332-I

All resources were cleaned up successfully. (S)

Datareplicator's shared resources have been deleted.

S: Terminates processing.

KFRB04333-I

There are no resources. (S)

There are no shared resources in Datareplicator.

S: Terminates processing.

KFRB04334-E

There is not host-name or server-name resource. (S)

There is no resource with the host name or server name specified in the *-x* option of the *hdeshmclean* command.

S: Cancels processing.

O: Check and, if necessary, revise the host name or server name specified in the `-x` option of the `hdeshmclean` command.

KFRB04335-E

Invalid combination of `startmode` operand in Synchronization reflection group. (S)

The combination of `startmode` operands among the synchronous import groups is invalid.

S: Cancels processing.

O: Correct the combination of the `startmode` operands among the synchronous import groups to either of the following, and then re-execute `hdsstart`:

(a) Use only `trn` and `tbl`.

(b) Use only `spd`.

KFRB04336-E

`Dsid` specified as Synchronization reflection group does not exist. (S)

The data linkage identifier specified for a synchronous import group does not exist.

S: Cancels processing.

O: Specify a data linkage identifier that was defined in a `dsidxxx` operand, and then initialize Datareplicator.

KFRB04337-E

Invalid `dsid` specified as Synchronization reflection group. (S)

The data linkage identifier specified for a synchronous import group is invalid.

S: Cancels processing.

O: Check if the specified data linkage identifier satisfies a condition listed below. If so, correct the data linkage identifier and perform initialization.

- A single character or a string of three to eight characters is specified.
- A character other than 0 to 9 and A to F is specified.

KFRB04338-E

Synchronization reflection group name does not exist. (S)

The specified synchronous import group name does not exist.

S: Cancels processing.

O: Check if the specified synchronous import group name satisfies a condition listed below. If so, correct the synchronous import group name and perform initialization.

- The synchronous import group name differs from the name specified in the

syncgroup001 operand.

- The syncgroup001 operand is not specified.

KFRB04339-E

Duplicate dsid specified as Synchronization reflection group.
(S)

A duplicate data linkage identifier was specified for a synchronous import group.

S: Cancels processing.

O: Check and, if necessary, revise the duplicate data linkage identifier. After correcting the data linkage identifier, perform initialization.

KFRB04343-W

Synchronous reflection group canceled, group name = *nn...n*. (S)

The synchronous import group was cancelled. Hereafter, the import transaction synchronization facility will not be supported for this synchronous import group.

nn...n: Synchronous import group name

S: Resumes processing.

KFRB04344-E

Invalid Synchronization reflection group name. (S)

The synchronous import group is invalid.

S: Cancels processing.

O: Correct the synchronous import group.

KFRB04345-E

Not specified dsid as Synchronization reflection group. (S)

Partial initialization is not supported for data linkage identifiers for synchronous import groups.

S: Cancels processing.

O: Specify a data linkage identifier that does not belong to a synchronous import group.

KFRB04400-E

HiRDB access error occurred, SQL kind = *aa...aa*, SQLCODE = *nn...nn*.
(S)

A HiRDB access error occurred.

aa...aa: Type of SQL access

nn...nn: SQLCODE

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about SQLCODE, see the manual *HiRDB Version 9 Messages*. If SQLCODE is -449, there might be multiple event control tables (hde_dtbl) in the source HiRDB. Delete any extra event control tables so that there is only one event control table in the source HiRDB.

KFRB04401-E

Invalid send system identifier, identifier = aa...aa. (S)

A target identifier is invalid.

aa...aa: Target identifier

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04402-E

Invalid server name, server name = aa...aa. (S)

A server name is invalid.

aa...aa: Server name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04410-E

Source site Datareplicator is already started. (S)

The source Datareplicator has already started.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04411-E

Source site Datareplicator is not started. (S)

The source Datareplicator has not been started.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04412-I

Source site Datareplicator initialization is interrupted. (S)

Source Datareplicator initialization processing has been cancelled.

S: Terminates processing.

KFRB04413-Q

Please confirm HiRDB is not running in replication mode and there is no need to use queue file recovery function, do you really want to continue initialization (y or n)? (S)

Check the following, and then enter whether you want initialization processing to proceed:

- Check whether the source HiRDB is using HiRDB Datareplicator.
- Check whether it is necessary to use the facility for recovering the extraction information queue file.

Entering *y*: Perform initialization processing.

Entering *n*: Cancel initialization processing and terminate the command.

If the source HiRDB is using HiRDB Datareplicator:

When *y* is entered, conformity between the source and target databases might be lost. In this case, you must initialize the data linkage environments at the source and target, and then re-create the target database.

If it is necessary to use the facility for recovering the extraction information queue file:

Enter *n*. If *y* is entered, the facility for recovering the extraction information queue file can no longer be used.

S: Waits for the user's reply.

O: To execute initial processing, enter *y*; to cancel initial processing, enter *n*.

KFRB04414-Q

Extracted data may be lost by initialization, do you really want to continue initialization (*y* or *n*)? (S)

Enter whether you want initialization processing to proceed, after you determine whether it is safe to execute initialization.

- When *y* is entered: Datareplicator executes initialization processing.
- When *n* is entered: Datareplicator cancels initialization processing.

S: Waits for the user's reply.

O: To execute initial processing, enter *y*; to cancel initial processing, enter *n*.

KFRB04415-E

SQL output processing can not start while extraction processing or transmission processing is working. (S + E)

Update-SQL output processing cannot start because extraction or transmission processing is underway.

S: Cancels processing.

O: Terminate the source Datareplicator, and then re-execute the command.

KFRB04416-E

Extraction processing or transmission processing can not start while SQL output processing is working. (S + E)

Extraction or transmission processing cannot start because update-SQL output processing is underway.

S: Cancels processing.

O: Terminate the source Datareplicator, and then re-execute the command.

KFRB04500-I

hdeprep command terminated normally. (S)

The hdeprep command terminated normally.

S: Terminates processing.

KFRB04501-E

Specified <table name> in <extract clause> in <extract statement> is not defined in HiRDB system, lineno = *aa...aa*, table name = *bb...bb.cc...cc*. (S)

The table name specified in the extract statement's extract clause is undefined in the HiRDB system.

aa...aa: Line number in the extraction definition file

bb...bb: Authorization identifier

cc...cc: Table identifier

S: Cancels processing.

O: Correct *authorization-identifier.table-identifier* and re-execute.

KFRB04502-E

Specified <column name> in <extract clause> in <extract statement> is not defined in HiRDB system, lineno = *aa...aa*, column name = *bb...bb*. (S)

The column name specified in the extract statement's extract clause is undefined in the HiRDB system.

aa...aa: Line number in the extraction definition file

bb...bb: Column name

S: Cancels processing.

O: Correct the column name or delete it, and then re-execute.

KFRB04503-E

Unable to specified <where clause> for forward matching log data name, lineno = *aa...aa*. (S)

The `where` clause is not permitted for an update information name with right truncation.

aa...aa: Line number in the extraction definition file

S: Cancels processing.

O: Specify the complete name of the update information in the `from` clause or delete the `where` clause, and then re-execute.

KFRB04504-E

Condition constant value is invalid, `lineno = aa...aa`. (S)

The value of the comparison constant is invalid in the send row selection condition.

aa...aa: Line number in the extraction definition file

S: Cancels processing.

O: Correct the value of the constant and re-execute.

KFRB04505-E

Number of condition in `<where clause>` exceeds maximum value. (S)

The number of send row selection conditions specified in the `where` clause exceeded the maximum (256).

S: Cancels processing.

O: Revise the send row selection conditions and re-execute.

KFRB04506-E

Unable to specified `<where clause>` for one of log data names from same table, `lineno = aa...aa`, log data name = `bb...bb`. (S)

The `where` clause cannot be specified for only one of the update information names in the same table.

aa...aa: Line number in the extraction definition file

bb...bb: Update information name

S: Cancels processing.

O: Revise the update information names subject to selection of send rows, and then re-execute.

KFRB04507-E

Condition column type is invalid, `lineno = aa...aa`, column name = `bb...bb`. (S)

The attribute of the send rows selection condition column is invalid.

aa...aa: Line number in the extraction definition file

bb...bb: Name of the send rows selection condition column

S: Cancels processing.

O: Revise the attribute of the send rows selection condition column, and then re-execute.

KFRB04508-E

Duplicate <send system identifier> in <extract control statement file>, send system id = *aa...aa*. (S)

A target identifier is duplicated in the extraction system definition.

aa...aa: Target identifier

S: Cancels processing.

O: Eliminate the duplicated target identifier, and then re-execute.

KFRB04509-E

Missing value for operand, operand name = *aa...aa*. (S)

An operand value is missing.

aa...aa: Operand name

S: Cancels processing.

O: Specify the required operand value, and then re-execute.

KFRB04510-E

Invalid sendidx operand in definition file, definition file = *aa...aa*. (S)

A sendidx operand is invalid.

aa...aa: Name of the extraction system definition

S: Cancels processing.

O: Correct the operand in the extraction system definition file and re-execute.

KFRB04511-E

<system table name> is specified in <extract clause> in <extract statement>, lineno = *aa...aa*. (S)

A data dictionary table cannot be subject to extraction processing.

aa...aa: Line number

S: Cancels processing.

O: Delete the extract statement and re-execute.

KFRB04512-E

Number of <extract statement> exceeds value of extinfunum operand. (S)

The number of extract statements exceeded the value specified in the extinfunum

operand in the extraction system definition file.

S: Cancels processing.

O: Correct so that the number of `extract` statements does not exceed the `extinforum` operand value, and then re-execute.

KFRB04513-E

Only mapping key column is specified in `<extract clause>` in `<extract statement>`, `lineno = aa...aa`. (S)

The extraction specification requires more than just the mapping key.

`aa...aa`: Line number in the extraction definition file

S: Cancels processing.

O: Revise the mapping key or delete the `extract` statement, and then re-execute.

KFRB04514-E

Specified user has no DBA privilege. (S)

The specified user does not have the DBA privilege.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04515-E

Number of `<mapping key>` exceeds maximum value, `lineno = aa...aa`. (S)

The number of specified mapping keys exceeded the maximum (16).

`aa...aa`: Line number in the extraction definition file

S: Cancels processing.

O: Revise the mapping keys and re-execute.

KFRB04516-E

`<event table name>` is specified in `<extract clause>` in `<extract statement>`, `lineno = aa...aa`. (S)

An event control table cannot be subject to extraction processing.

`aa...aa`: Line number in the extraction definition file

S: Cancels processing.

O: Delete the `extract` statement and re-execute.

KFRB04517-E

Invalid `<log data name>` in `<to clause>` in `<extract statement>`, `lineno = aa...aa`, `log data name = bb...bb`. (S)

The update information name is invalid in the `extract` statement's `to` clause.

aa...aa: Line number in the extraction definition file

bb...bb: Update information name

S: Cancels processing.

O: Correct the update information name and re-execute.

KFRB04518-E

Invalid <log data name> in <from clause> in <send statement>, lineo = *aa...aa*, log data name = *bb...bb*. (S)

The update information name is invalid in the send statement's from clause.

aa...aa: Line number in the extraction definition file

bb...bb: Update information name

S: Cancels processing.

O: Correct the update information name and re-execute.

KFRB04519-E

Invalid <schema name> in <extract clause> in <extract statement>, lineo = *aa...aa*, schema name = *bb...bb*. (S)

The schema name is invalid in the extract statement's extract clause.

aa...aa: Line number in the extraction definition file

bb...bb: Schema name

S: Cancels processing.

O: Correct the schema name and re-execute.

KFRB04520-E

Invalid <table name> in <extract clause> in <extract statement>, lineo = *aa...aa*, table name = *bb...bb*. (S)

The table name is invalid in the extract statement's extract clause.

aa...aa: Line number in the extraction definition file

bb...bb: Table name

S: Cancels processing.

O: Correct the table name and re-execute.

KFRB04521-E

Invalid <column name> in <extract clause> in <extract statement>, lineo = *aa...aa*, column name = *bb...bb*. (S)

The column name is invalid in the extract statement's extract clause.

aa...aa: Line number in the extraction definition file

bb...bb: Column name

S: Cancels processing.

O: Correct the column name or delete it, and then re-execute.

KFRB04522-E

Invalid <send system identifier> in <send clause> in <send statement>, lineno = *aa...aa*, send system identifier = *bb...bb*. (S)

The target identifier is invalid in the send statement's send clause.

aa...aa: Line number in the extraction definition file

bb...bb: Target identifier

S: Cancels processing.

O: Correct the target identifier and re-execute.

KFRB04523-E

Specified table name is not base table name, lineno = *aa...aa*, table name = *bb...bb*. (S)

Specified table is not a base table.

aa...aa: Line number in the extraction definition file

bb...bb: Table name

S: Cancels processing.

O: Specify a base table's name or column name and re-execute.

KFRB04524-E

Syntax error in <extract definition file>, lineno = *aa...aa*. (S)

There is a syntax error in the extraction definition file.

aa...aa: Line number in the extraction definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04525-E

Number of <extract statement> exceeds maximum value. (S)

The number of extract statements exceeded the maximum.

S: Cancels processing.

O: Reduce the number of extract statements and re-execute.

KFRB04526-E

Number of <column name> exceeds maximum value. (S)

The number of specified column names exceeded the maximum.

S: Cancels processing.

O: Reduce the number of columns subject to extraction, and then re-execute.

KFRB04527-E

Invalid <mapping key> column length in <key clause> in <extract statement>, lineno = *aa...aa*, column name = *bb...bb*. (S)

At least one of the columns constituting the mapping key exceeds the maximum length for a mapping key (256 bytes).

aa...aa: Line number in the extraction definition file

bb...bb: Column name

S: Cancels processing.

O: Eliminate the corresponding column from the mapping key and re-execute.

KFRB04528-E

Duplicate <mapping key> column in <key clause> in <extract statement>, lineno = *aa...aa*, column name = *bb...bb*. (S)

A mapping key column name is duplicated in the `extract` statement's key clause.

aa...aa: Line number in the extraction definition file

bb...bb: Column name

S: Cancels processing.

O: Eliminate the duplication and re-execute.

KFRB04529-E

<mapping key> column is not specified in <extract clause> in <extract statement>, lineno = *aa...aa*, column name = *bb...bb*. (S)

No mapping key column is specified in the `extract` statement's `extract` clause.

aa...aa: Line number in the extraction definition file

bb...bb: Column name

S: Cancels processing.

O: Define a mapping key column (unique column) and re-execute.

KFRB04530-E

<send system identifier> in <send clause> in <send statement> does not define in <extract control statement file>, lineno = *aa...aa*, send system identifier = *bb...bb*. (S)

The target identifier specified in the `send` statement's `send` clause is not defined in the extraction system definition.

aa...aa: Line number in the extraction definition file

bb...bb: Target identifier

S: Cancels processing.

O: Correct the target identifier and re-execute.

KFRB04531-E

Number of `<send statement>` exceeds maximum value. (S)

The number of `send` statements exceeded the maximum.

S: Cancels processing.

O: Reduce the number of `send` statements and re-execute.

KFRB04532-W

Event table is not defined. (S)

No event control table has been defined.

S: Resumes processing. However, Datareplicator cannot issue events.

O: To use events, create an event control table, and then re-execute the `hdeprep` command.

KFRB04533-E

Specified `<log data name>` is not defined in `<extract statement>`,
`lineno = aa...aa`, `log data name = bb...bb`. (S)

The specified update information name is not defined with the `extract` statement.

aa...aa: Line number in the extraction definition file

bb...bb: Update information name

S: Cancels processing.

O: Add an `extract` statement or correct the update information name, and then re-execute.

KFRB04534-E

Duplicate `<log data name>` in `<to clause>` in `<extract statement>`,
`lineno = aa...aa`, `log data name = bb...bb`. (S)

Update information is duplicated in the `extract` statement's `to` clause.

aa...aa: Line number in the extraction definition file

bb...bb: Update information name

S: Cancels processing.

O: Delete the duplicated name or modify the specification, and then re-execute.

KFRB04535-E

Event table must be a `fix` table. (S)

The event control table is not defined as a table with the FIX attribute.

S: Cancels processing.

O: Re-create the event control table (by executing DROP TABLE and CREATE TABLE) and re-execute.

KFRB04536-E

Invalid column numbers of event table. (S)

The number of event control table columns is invalid.

S: Cancels processing.

O: Reevaluate the structure of the event control table, re-create the event control table (by executing DROP TABLE and CREATE TABLE), and then re-execute.

KFRB04537-E

Invalid column of event table. (S)

A column's attributes (data type, length, name) are invalid for an event control table.

S: Cancels processing.

O: Reevaluate the structure of the event control table, re-create the event control table (by executing DROP TABLE and CREATE TABLE), and then re-execute.

KFRB04538-E

Unable to decide target system for <log data name>, log data name = *aa...aa*. (S)

Datareplicator was unable to determine the target system for the update information name.

aa...aa: Update information name

S: Cancels processing.

O: Reevaluate the correspondence to the target system, correct it, and then re-execute.

KFRB04539-E

Unsupported data type was found, lineno = *aa...aa*, column name = *bb...bb*. (S)

This data type is not supported for extraction processing.

aa...aa: Line number in the extraction definition file

bb...bb: Column name

S: Cancels processing.

O: Delete the corresponding column from the definition and re-execute.

KFRB04540-E

Extraction master process not exist. (S)

The extraction master process was not found.

S: Cancels processing.

O: Start the extraction Datareplicator and re-execute.

KFRB04541-E

Condition column length is invalid, lineno = *aa...aa*, column name = *bb...bb*. (S)

The length of the send row selection condition column is invalid.

aa...aa: Line number in the extraction definition file

bb...bb: Name of the send row selection condition column

S: Cancels processing.

O: Reevaluate the length of the send row selection condition column, correct it, and then re-execute.

KFRB04542-E

Number of constant value in <in condition> exceeds maximum value, lineno = *aa...aa*. (S)

The number of constant values specified in the in condition exceeded the maximum (16).

S: Cancels processing.

O: Reevaluate the number of constant values, correct it, and then re-execute.

KFRB04543-E

Constant value length is invalid, lineno = *aa...aa*. (S)

The length of a comparison constant is invalid in the send row selection condition column.

aa...aa: Line number in the extraction definition file

S: Cancels processing.

O: Reevaluate the length of the constant value, correct it, and then re-execute.

KFRB04544-E

Constant value type is invalid, lineno = *aa...aa*. (S)

The attribute of a comparison constant is invalid in the send row selection condition column.

aa...aa: Line number in the extraction definition file

S: Cancels processing.

O: Reevaluate the length of the constant value, correct it, and then re-execute.

KFRB04545-E

Condition column is not mapping key, lineno = *aa...aa*, column name = *bb...bb*. (S)

The send row selection condition column is not the mapping key.

aa...aa: Line number in the extraction definition file

bb...bb: Name of the send row selection condition column

S: Cancels processing.

O: Reevaluate the send row selection condition column, correct it, and then re-execute.

KFRB04546-E

Not found condition column in log data, lineno = *aa...aa*, column name = *bb...bb*. (S)

There is no send row selection condition column in the update information.

aa...aa: Line number in the extraction definition file

bb...bb: Name of the send row selection condition column

S: Cancels processing.

O: Reevaluate the send row selection condition column, correct it, and then re-execute.

KFRB04547-E

Constant value is larger than condition column value range, lineno = *aa...aa*. (S)

The comparison constant in the send row selection condition is outside the range of permissible values for the condition column.

aa...aa: Line number in the extraction definition file

S: Cancels processing.

O: Reevaluate the constant value, correct it, and then re-execute.

KFRB04548-E

Duplicate <send system identifier> / <log data name> in <send statement>, lineno = *aa...aa*, send system identifier = *bb...bb*, log data name = *cc...cc*. (S)

The combination of target identifier and update information name is duplicated in the send statement.

aa...aa: Line number in the extraction definition file

bb...bb: Target identifier

cc...cc: Update information name

S: Cancels processing.

O: Reevaluate the combination of target identifier and update information name in the send statement, correct it, and then re-execute.

KFRB04549-E

Unable to specified <flike condition> for numerical column,
lineno = *aa...aa*, column name = *bb...bb*. (S)

The flike condition is not permitted for a numeric column.

aa...aa: Line number in the extraction definition file

bb...bb: Name of the send row selection condition column

S: Cancels processing.

O: Reevaluate the send row selection condition column, correct it, and then re-execute.

KFRB04550-E

Starting offset in <flike condition> is invalid, lineno = *aa...aa*.
(S)

The comparison start offset value is invalid in the flike condition.

aa...aa: Line number in the extraction definition file

S: Cancels processing.

O: Reevaluate the comparison start offset, correct it, and then re-execute.

KFRB04551-E

Starting offset in <flike condition> is larger than condition
column definition length, lineno = *aa...aa*. (S)

The comparison start offset value in the flike condition exceeds the length defined
for the condition column.

aa...aa: Line number in the extraction definition file

S: Cancels processing.

O: Reevaluate the comparison start offset, correct it, and then re-execute.

KFRB04552-E

Recovery type of BLOB column is not 'all', lineno = *aa...aa*, column
name = *bb...bb*. (S)

BLOB column's column recovery restriction is not ALL.

aa...aa: Line number in the extraction definition file

bb...bb: Column name subject to extraction processing

S: Cancels processing.

O: Change the column recovery restriction to ALL for the column subject to extraction
processing, and then re-execute.

KFRB04553-E

BLOB column length is invalid, lineno = *aa...aa*, column name = *bb...bb*. (S)

BLOB column's definition length is invalid.

aa...aa: Line number in the extraction definition file

bb...bb: Column name subject to extraction processing

S: Cancels processing.

O: Reevaluate the definition length of the column subject to extraction processing, correct it, and then re-execute.

KFRB04554-E

Unable to specified BLOB column for mapping key, lineno = *aa...aa*, column name = *bb...bb*. (S)

A BLOB column is not permitted in the mapping key.

aa...aa: Line number in the extraction definition file

bb...bb: Column name subject to extraction processing

S: Cancels processing.

O: Reevaluate the mapping key column, correct it, and then re-execute.

KFRB04555-E

Unable to specified Abstract Data Type column for mapping key, lineno = *aa...aa*, column name = *bb...bb*. (S)

An abstract data type column is not permitted for the mapping key.

aa...aa: Line number in the extraction definition file

bb...bb: Column name subject to extraction processing

S: Cancels processing.

O: Reevaluate the mapping key column, correct it, and then re-execute.

KFRB04556-E

Number of <adt statement> exceeds maximum value. (S)

The number of adt statements exceeded the maximum.

S: Cancels processing.

O: Reduce the number of adt statements and re-execute.

KFRB04557-E

Invalid <schema name> in <adt clause> in <adt statement>, lineno = *aa...aa*, schema name = *bb...bb*. (S)

The schema name is invalid in the adt statement's adt clause.

aa...aa: Line number in the extraction definition file

bb...bb: Schema name

S: Cancels processing.

O: Correct the schema name and re-execute.

KFRB04558-E

Invalid <data type name> in <adt clause> in <adt statement>,
lineno = *aa...aa*, data type name = *bb...bb*. (S)

The data type is invalid in the adt statement's adt clause.

aa...aa: Line number in the extraction definition file

bb...bb: Data type name

S: Cancels processing.

O: Correct the data type and re-execute.

KFRB04559-E

Invalid <UOC name> in <by clause> in <adt statement>, lineno =
aa...aa, UOC name = *bb...bb*. (S)

The UOC routine name is invalid in the adt statement's by clause.

aa...aa: Line number in the extraction definition file

bb...bb: UOC routine name

S: Cancels processing.

O: Correct the UOC routine name and re-execute.

KFRB04560-E

Invalid <UOC library name> in <lib clause> in <adt statement>,
lineno = *aa...aa*, UOC library name = *bb...bb*. (S)

The UOC library name is invalid in the adt statement's lib clause.

aa...aa: Line number in the extraction definition file

bb...bb: UOC library name

S: Cancels processing.

O: Correct the UOC library name and re-execute.

KFRB04561-E

Not exist UOC library specified by <adt statement>, lineno =
aa...aa, UOC library name = *bb...bb*. (S)

The UOC library specified in the adt statement was not found.

aa...aa: Line number in the extraction definition file

bb...bb: UOC library name

S: Cancels processing.

O: Create the UOC library, and then re-execute.

KFRB04562-E

Not exist UOC function into UOC library, lineno = *aa...aa*, UOC function name = *bb...bb*, UOC library name = *cc...cc*. (S)

A UOC function was not found in the UOC library.

aa...aa: Line number in the extraction definition file

bb...bb: UOC function name

cc...cc: UOC library name

S: Cancels processing.

O: Re-create the UOC library and re-execute.

KFRB04563-E

Specified data type is not nest unit Abstract data type of extract target, lineno = *aa...aa*, schema name = *bb...bb*, data type name = *cc...cc*. (S)

Specified data type is not a nest-based abstract data type subject to extraction processing.

aa...aa: Line number in the extraction definition file

bb...bb: Schema name

cc...cc: Data type name

S: Cancels processing.

O: Correct the data type and re-execute.

KFRB04564-E

Duplicate <adt statement> for same data type, lineno = *aa...aa*, schema name = *bb...bb*, data type name = *cc...cc*. (S)

The adt statement is duplicated for the same data type.

aa...aa: Line number in the extraction definition file

bb...bb: Schema name

cc...cc: Data type name

S: Cancels processing.

O: Correct the duplication of adt statements and re-execute.

KFRB04565-E

<adt statement> is not defined for this Abstract data type,
schema name = *aa...aa*, data type name = *bb...bb*. (S)

No adt statement is defined for this abstract data type.

aa...aa: Schema name

bb...bb: Data type name

S: Cancels processing.

O: Define an adt statement for every abstract data type and re-execute.

KFRB04566-E

Unsupported Abstract data type was found, lineno = *aa...aa*, type
name = *bb...bb*. (S)

Datareplicator cannot extract this abstract data type.

aa...aa: Line number in the extraction definition file

bb...bb: Name of the abstract data type

S: Cancels processing.

O: Delete the corresponding abstract data type from the extraction definition and
re-execute.

KFRB04567-E

Unable to specify <adt statement> for this Abstract data type,
lineno = *aa...aa*, type name = *bb...bb*. (S)

The adt statement is not permitted for this abstract data type.

aa...aa: Schema name

bb...bb: Data type name

S: Cancels processing.

O: Delete the unneeded adt statement and re-execute.

KFRB04568-E

Unable to extract multi-value column, lineno = *aa...aa*, column name
= *bb...bb*. (S)

Datareplicator was unable to extract a repetition column.

aa...aa: Line number in the extraction definition file

bb...bb: Column name subject to extraction processing

S: Cancels processing.

O: Delete the repetition column from the columns subject to extraction processing or
change the HiRDB version subject to extraction processing to 05-05 or later, and then

re-execute.

KFRB04569-E

Unable to specify multi-value column for mapping key, lineno = *aa...aa*, column name = *bb...bb*. (S)

A repetition column is not permitted for the mapping key.

aa...aa: Line number in the extraction definition file

bb...bb: Column names subject to extraction processing

S: Cancels processing.

O: Reevaluate the mapping key columns, correct the error, and then re-execute.

KFRB04570-E

Unable to specify each data type for same column, lineno = *aa...aa*, table name = *bb...bb*, column name = *cc...cc*. (S)

You cannot specify different data types for the same column.

aa...aa: Line number in the extraction definition file

bb...bb: Table name subject to extraction processing

cc...cc: Column name subject to extraction processing

S: Cancels processing.

O: Eliminate the invalid specification of extraction data type for the same column, and then re-execute.

KFRB04572-E

Unable to specified BINARY column for mapping key, lineno = *aa...aa*, column name = *bb...bb*. (S)

A BINARY-attribute column cannot be specified for a mapping key.

aa... aa: Line number in the extraction definition file

bb...bb: Name of column to be extracted

S: Cancels processing.

O: Check the mapping key column for errors. If there is an error, correct it, and then re-execute.

KFRB04573-E

Failed to get information of unique check of mapping key, SQLCODE = *aa...aa*, SQL = *bb...bb*. (S)

Datareplicator was unable to obtain information needed for unique check of mapping key column.

aa... aa: SQLCODE

bb...bb: SQL statement

S: Cancels processing.

O: Check the detail information, eliminate the cause of the error, and then re-execute the command.

KFRB04574-E

Mapping key column does not satisfy condition of unique check, table name = *aa...aa*, mapping key = *bb...bb*, code = *cc...cc*. (S)

The mapping key column does not satisfy the conditions of the unique check.

aa...aa: Source table name

bb...bb: Name of mapping key component column

cc...cc: Cause code

UNIQUE: An index satisfying the conditions of the unique check has not been defined for the source table.

NOT NULL: The index satisfying the conditions of the unique check includes a column whose attribute is not NOT NULL.

S: Cancels processing.

O: Check the mapping key column for errors. If there is an error, correct the definition. If there is no error in the specification, take appropriate action on the basis of the cause code. If you cannot take such action, change the value of the check clause in the extraction definition or the value of the *-k* option during command execution, and then re-execute the *hdeprep* command.

- UNIQUE

For the source table, define an index that satisfies the conditions of the unique check.

- NOT NULL

Changes the component column of the index satisfying the conditions of the unique check to NOT NULL attribute.

KFRB04575-E

Extract data size exceeds maximum value, lineno = *aa...aa*, log data name = *bb...bb*. (S + L)

Update information exceeded the maximum size.

aa...aa: Line number

bb...bb: Name of update information

S: Cancels processing.

O: Correct the extraction definition so that the update information does not exceed the maximum size (256 megabytes), and then re-execute the command.

KFRB04579-E

Invalid extraction environment for extraction table that is specified without rollback option, `lineno = aa...aa`, `reason = bb...bb`. (S)

A condition for using a table for which the `WITHOUT ROLLBACK` option is specified as the source table is not satisfied.

`aa...aa`: Line number in the extraction definition file

`bb...bb`: Reason code

S: Stops processing.

[Action] Take the following action according to the displayed reason code:

Reason code	Description	Action
<code>ukey</code>	A column specified for a mapping key cannot be updated in a table for which the <code>WITHOUT ROLLBACK</code> option is specified. Therefore, the <code>ukey</code> clause cannot be specified in the extraction definition statement.	Change the specification of the key clause in the extraction definition statement, and then re-execute the command.
<code>sendcontrol</code>	If a table for which the <code>WITHOUT ROLLBACK</code> option is specified is used as the source table, <code>sendmst</code> cannot be specified in the <code>sendcontrol</code> operand in the extraction system definition.	Change the <code>sendcontrol</code> operand value to <code>nodemst</code> (default) in the extraction system definition, initialize Datareplicator with the <code>hdestart -i</code> command, and then re-execute the command.
<code>divided table</code>	A table that is row-partitioned among multiple servers and for which the <code>WITHOUT ROLLBACK</code> option is specified cannot be used as the source table.	Define the table without the <code>WITHOUT ROLLBACK</code> option specified, and then re-execute the command.

KFRB04580-E

Without rollback option is unable to specify for event table. (S)

The `WITHOUT ROLLBACK` option cannot be specified for an event control table (`hde_dtbl`).

S: Stops processing.

[Action] Define the event control table without the `WITHOUT ROLLBACK` option specified, and then re-execute the command.

KFRB04601-E

Duplicate <dataset name> in <restruct clause> in <restruct statement> or in <extract clause> in <extract statement>, lineno = *nn...nn*, dataset name = *aa...aa*. (S+L)

A dataset name is duplicated in the `restruct` or `extract` clause.

nn...nn: Line number in the update information definition file

aa...aa: Dataset name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04602-E

Duplicate <field name> in <field clause> in <restruct statement> or in <extract statement> in <extract statement>, lineno = *nn...nn*, field name = *aa...aa*. (S+L)

A field name is duplicated in the `field` or `extract` clause.

nn...nn: Line number in the update information definition file

aa...aa: Field name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04603-E

Invalid field data position in <position clause> in <restruct statement>, lineno = *nn...nn*. (S+L)

The value specified in the `position` clause for the data start position is invalid.

nn...nn: Line number in the update information definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04604-E

Invalid field type length in <attr clause> in <restruct statement>, lineno = *nn...nn*. (S+L)

The value specified in the `attr` clause for the length of the redefined field attribute is invalid.

nn...nn: Line number in the update information definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04605-E

Invalid <DBM name> in <restruct clause> in <restruct statement> or in <extract clause> in <extract statement>, lineno = *nn...nn*, DBM name = *aa...aa*. (S+L)

DBM name in the `restruct` or `extract` clause is invalid.

nn...nn: Line number in the update information definition file

aa...aa: DBM name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04606-E

Invalid <dataset name> in <restruct clause> in <restruct statement> or in <extract clause> in <extract statement>, lineno = *nn...nn*, dataset name = *aa...aa*. (S+L)

A dataset name in the `restruct` or `extract` clause is invalid.

nn...nn: Line number in the update information definition file

aa...aa: Dataset name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04607-E

Invalid <field name> in <field clause> in <restruct statement>, lineno = *nn...nn*, field name = *aa...aa*. (S+L)

A field name in the `field` clause is invalid.

nn...nn: Line number in the update information definition file

aa...aa: Field name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04608-E

Cannot specify asterisk as all columns in <extract clause> in <extract statement>, lineno = *nn...nn*. (S+L)

Asterisk is not allowed in the `extract` clause.

nn...nn: Line number in the update information definition file

S: Cancels processing.

O: Specify a field name and re-execute.

KFRB04609-E

Invalid uoc file name length in <uocname statement>, lineno = *nn...nn*. (S+L)

UOC filename is invalid.

nn...nn: Line number in the update information definition file

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04610-E

Cannot specify not null option in <attr clause> in <restruct statement>, lineno = *nn...nn*, field name = *aa...aa*. (S+L)

The attr operand's not null option is not permitted.

nn...nn: Line number in the update information definition file

aa...aa: Field name

S: Cancels processing.

O: Delete the not null option from the line resulting in the error, and then re-execute.

KFRB04611-E

Specified dsid is not for hdssamqin command, dsid = *xx*. (S+L)

The type of data linkage identifier is invalid in the update information input command (hdssamqin command).

xx: Data linkage identifier

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04612-E

hdssamqin command is already in use. (S+L)

The update information input command (hdssamqin command) is already in use.

S: Cancels processing.

O: Re-execute after the current command has terminated. If no command is executing, the previous command might have terminated abnormally. In this case, execute the command with the -o option specified.

KFRB04613-E

Invalid update kind in log data file, read position = *aa...aa*, update kind = *bb...bb*. (S+L)

The update type is invalid in the update information file.

aa...aa: Update data read position

bb...bb: Update type

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04614-E

Failed to convert numeric data, read position = *aa...aa*, field name = *bb...bb*, error code = *nn...nn*. (S+L)

Conversion of numeric data failed.

aa...aa: Update data read position

bb...bb: Field name resulting in the conversion error

nn...nn: Error code:

1: Data length overflow

2: Invalid data value

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04615-E

Update queue file write error, *errno* = *xx...xx*. (S+L)

A write error occurred in the import information queue file.

xx...xx: Error number set in *errno*

S: Cancels processing.

O: Check *errno.h* or the user's OS documentation for the error number, eliminate the cause of the error, and then re-execute.

KFRB04616-E

Invalid kind of queue data, error code = *nn...nn*. (S+L)

Data is invalid in the queue input file.

nn...nn: Error code:

1: The file does not begin with the port information (it is not queue data).

2: The data linkage identifier does not match.

3: The source DBMS is invalid.

4: The transmission sequence ID is invalid.

5: The extraction definition information was not found.

6: The update information was not found.

7: The update information split code is invalid.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04617-E

Internal error, function name = *aa...aa*, code = *nn*. (S+L)

An internal error occurred.

aa...aa: Function name

nn: Termination code

S: Cancels processing.

O: Contact the customer support center.

KFRB04618-I

Log data file replication completed successfully. (S+L)

Input of the update information file has been completed.

S: Terminates processing.

KFRB04619-E

Invalid data start position, DBM name = *aa...aa*, dataset name = *bb...bb*, field name = *cc...cc*. (S+L)

The data start position is invalid.

aa...aa: DBM name

bb...bb: Dataset name

cc...cc: Field name

S: Cancels processing.

O: Reevaluate the `restruct` statement in the update information definition file, eliminate the cause of the error, and then re-execute.

KFRB04620-E

`hdssamqin` command caught signal *nn...nn*. (S+L)

The update information input command received a signal.

nn...nn: Received signal

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04621-E

First log data is not 'PH' log. (S+L)

The first log is not the log header information (log ID: PH).

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04622-E

Invalid runid or file number of log data file, the last time runid = *xxxx*, file number = *mm...mm*, the present runid = *yyyy*, file number = *nn...nn*, endinfo = *aa...aa*. (S+L)

The run ID or file sequence number is invalid.

xxxx: Last run ID

mm...mm: Last file number

yyyy: Current run ID

nn...nn: Current file number

aa...aa: Termination information about the previous execution

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04623-E

Used PDMII user change routine. (S+L)

Datereplicator cannot read update information data using PDM2's user conversion routine.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04624-E

Failed to get update data from PDMII log file, read position = *aa...aa*, field name = *bb...bb*, field position = *mm...mm*, field length = *nn...nn*, total data size = *xx...xx*. (S+L)

Datereplicator was unable to obtain update data from PDM2 E2's log file.

aa...aa: Update data read position

bb...bb: Field name

mm...mm: Field position

nn...nn: Field length

xx...xx: Total data size

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04625-E

No dataset name in 'ED' log, DBM name = *aa...aa*, dataset name = *bb...bb*. (S+L)

A data set name specified in the update information definition file was not found in the update extraction definition information (ED log).

aa...aa: DBM name

bb...bb: Dataset name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04626-E

No DBM name in 'ED' log, DBM name = *aa...aa*. (S+L)

DBM name specified in the update information definition file was not found in the update extraction definition information (ED log).

aa...aa: DBM name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04627-E

Invalid log data. (S+L)

Log data is invalid. Update log information was output before the update extraction definition information (ED log) was obtained.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04628-E

Duplicate 'ED' log data. (S+L)

The update extraction definition information (ED log) has already been obtained.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04629-E

No 'ED' log data. (S+L)

The update extraction definition information (ED log) has not been obtained yet.

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04630-E

Invalid mapping key field name, field name = *aa...aa*. (S+L)

A mapping key field name is invalid in the update information definition file.

aa...aa: Field name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04631-E

Update queue file is full. (S+L)

The import information queue file is full.

S: Cancels processing.

O: Re-execute when the import information queue file becomes available as a result of execution of import processing.

KFRB04632-I

Log data file read information, read position = *nn...nn*. (S+L)

This message provides information about the read position in the update information file.

nn...nn: Read position (import position in the import information queue file)

S: Cancels processing.

O: See the previous error message, eliminate the cause of the error, and then re-execute.

KFRB04633-E

Extract field is all mapping key, DBM name = *aa...aa*, dataset name = *bb...bb*. (S+L)

The extraction fields specified in the update information definition file are all part of the mapping key.

aa...aa: DBM name

bb...bb: Dataset name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04634-E

Name data convert error occurred, *aa...aa* name = *bb...bb*, code = *nn...nn*. (S+L)

A code conversion error occurred on the DBM name, dataset name, field name, or update information name.

aa...aa: Type of name resulting in the error:

DBM: DBM name

dataset: Dataset name

field: Field name

update: Update information name

bb...bb: Name resulting in the error

nn...nn: Conversion error code

S: Cancels processing.

O: Correct the erroneous name and re-execute.

KFRB04635-E

Mapping key field is not a lowest lank field, DBM name = *aa...aa*, dataset name = *bb...bb*, mapping key field = *cc...cc*. (S+L)

A mapping key field specified in the `extract` clause in the update information definition file is not at the lowest field level.

aa...aa: DBM name

bb...bb: Dataset name

cc...cc: Name of the mapping key field

S: Cancels processing.

O: Correct the update information definition file and re-execute.

KFRB04636-E

No extract field name in 'ED' log, DBM name = *aa...aa*, dataset name = *bb...bb*, field name = *cc...cc*. (S+L)

A field name specified in the `extract` clause in the update information definition file was not found in the update extraction definition information (ED log).

aa...aa: DBM name

bb...bb: Dataset name

cc...cc: Field name

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04637-I

Unextracted data is in output file, file name = *aa...aa*. (S+L)

Datareplicator output unextracted data from the update information input file to the unextracted information file.

aa...aa: Name of the unextracted information file

S: Cancels processing.

O: To import the update information that was output to the unextracted information file, create a definition specifying only the dataset name corresponding to the

unextracted information in the update information definition file, and then re-execute the `hdssamqin` command with the `-c` option specified.

KFRB04638-E

Field length too large for reflect column, DBM name = *aa...aa*, dataset name = *bb...bb*, field name = *cc...cc*. (S+L)

A field length exceeds the maximum permitted for an import column.

aa...aa: DBM name

bb...bb: Dataset name

cc...cc: Field name

S: Cancels processing.

O: Correct the update information definition file and re-execute the command.

KFRB04639-E

Invalid use of length information field, DBM name = *aa...aa*, dataset name = *bb...bb*, field name = *cc...cc*. (S+L)

You cannot specify a record length storage field for variable-length data in the extraction statement.

aa...aa: DBM name

bb...bb: Dataset name

cc...cc: Field name

S: Cancels processing.

O: Correct the update information definition file and re-execute the command.

KFRB04640-E

Invalid kind of log data file. (S+L)

The type of update information file is invalid.

S: Cancels processing.

O: Re-execute the command with the `-c` option specified.

KFRB04641-E

DBMS license check error, status = *aa...aa*. (S+L)

A DBMS license check error occurred.

aa...aa: Detail code

S: Cancels processing.

O: Take appropriate action based on the following table, and then re-execute the command:

Detail code	Description	Action
100	HiRDB Datareplicator Extension has not been installed.	Install the HiRDB Datareplicator Extension and re-execute.
101 102	The license file is damaged.	Reinstall the HiRDB Datareplicator Extension.
103	No license was required for establishing connection with the other company's database.	Install the license product and re-execute.
104	The license required for establishing connection with the other company's database has expired.	Contact the system administrator.

KFRB04642-E

Invalid log data file, file name = *aa...aa*. (S+L)

The update information file is invalid.

aa...aa: Name of the update information file

S: Cancels processing.

O: Re-create the update information file and re-execute the command.

KFRB04643-W

Insufficient buffer occurred while editing log data, record size = *aa...aa*. (S+L)

The update information editing buffer is insufficient.

aa...aa: Update record length

S: Cancels processing.

O: Specify a size for the update information editing buffer that is at least the update record length in the `hdssamqin` command's `-l` option, and then re-execute the command.

KFRB04701-E

Unable to get directory path list, function = *aa...aa*, errno = *bb...bb*. (S+L)

An error occurred while obtaining a list of information subject to monitoring from the operation directory file.

aa...aa: Name of the function resulting in the error

bb...bb: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB04702-E

Invalid operation in MIB file. operation = *aa...aa*. (S+L)

A command is invalid in the MIB file.

aa...aa: Operation specified in the MIB file

S: Cancels processing.

O: The contents of the MIB file might have been updated at the machine, resulting in the error. Restore the MIB file to its status when Datareplicator was installed, and then re-execute.

KFRB04703-E

Memory allocate error, errno = *aa...aa*, size = *bb...bb*. (L)

Datareplicator was unable to allocate a memory area while collecting information subject to monitoring.

aa...aa: Error number set in `errno`

bb...bb: Area allocation size

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB04704-E

Shared memory access error, path = *aa...aa*, function = *bb...bb*, errno = *cc...cc*. (L)

A shared memory access error occurred while collecting information subject to monitoring.

aa...aa: Name of operation directory

bb...bb: Name of function resulting in the error

cc...cc: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. For details about the error number, see `errno.h` or the OS documentation.

KFRB04705-W

Unable to access shared memory for exclusion, path = *aa...aa*. (L)

Datareplicator was unable to collect Datareplicator information due to another command's lock control.

aa...aa: Name of the operation directory

S: Skips this Datareplicator information and collects the next Datareplicator information.

KFRB04706-E

Remote control operation format error, operation = *aa...aa*. (L)

The format of remote control operation that was specified by the MIB browser is invalid.

aa...aa: Remote control operation specified by the MIB browser

S: Cancels processing.

O: Check the format of the remote control operation specified by the MIB browser's SNMP value, and then re-execute.

KFRB04707-E

Remote control operation index error, index = *aa...aa*. (L)

The index value of remote control operation that was specified by the MIB browser is invalid.

aa...aa: Datareplicator's index subject to remote control specified by the MIB browser

S: Cancels processing.

O: Check the index of the remote control operation specified by the MIB browser's SNMP value, and then re-execute.

KFRB04708-I

Remote control operation start, operation = *aa...aa*, *bb...bb=cc...cc*. (L)

Datareplicator is starting remote control processing specified by the MIB browser.

aa...aa: Datareplicator operation subject to remote control specified by the MIB browser

bb...bb: Datareplicator type:

HDEPATH: Source Datareplicator

HDSPATH: Target Datareplicator

cc...cc: Name of operation directory

S: Resumes processing.

KFRB04709-E

Unable to execute remote control operation, command name = *aa...aa*,
errno = *bb...bb*. (L)

Datareplicator was unable to execute remote control processing specified by the MIB

browser.

aa...aa: Name of the command for remote control operation specified by the MIB browser

bb...bb: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04751-E

Duplicate <directory path> in <directory file>, directory path = *aa...aa*. (S)

The operation directory to be added to the objects of monitoring has already been registered.

aa...aa: Specified operation directory name

S: Cancels processing.

O: Check the registered operation directories and re-execute.

KFRB04752-E

Not exist <directory path> in <directory file>, directory path = *aa...aa*. (S)

The operation directory to be deleted from the objects of monitoring is not registered.

aa...aa: Specified operation directory name

S: Cancels processing.

O: Check the registered operation directories and re-execute.

KFRB04753-E

Not exist authorization to access <directory file>. (S)

The user does not have access privilege for the operation directory file.

S: Cancels processing.

O: Have the `root` user execute the command.

KFRB04801-E

command format error. (S)

Format of the Datareplicator file system area initialization or display command is invalid.

S: Cancels processing.

O: Re-execute the command in the correct format.

KFRB04802-E

Invalid -l operand, file number must be 1 - 255. (S)

The specified operand (-l) is invalid. The permitted number of files is in the range 1 to 255.

S: Cancels processing.

O: Check the operand (-l) and re-execute.

KFRB04803-E

Invalid -f operand, max file path length must be under 127 bytes.
(S)

The specified operand (-f) is invalid. The Datareplicator file system name must be expressed as 1 to 127 characters.

S: Cancels processing.

O: Check the operand (-f) and re-execute.

KFRB04804-Q

aa...aa = *bb...bb* is active.

Force initialization for file system area, do you want to continue initialization (y or n)? (S)

Asks if you want to execute forced initialization of the Datareplicator file system area. To execute, enter y; otherwise, enter n.

This message is displayed only when -r force is specified in the hdsfmkfs command.

aa...aa: HDEPATH for the source system and HDSPATH for the target system

bb...bb: Name of the Datareplicator directory

S: Waits for the user's reply.

O: To execute forced initialization, enter y; otherwise, enter n.

KFRB04805-E

Specified system file area is not initialized, please initialize by hdsfmkfs command, file system area = *aa...aa*. (S)

The specified Datareplicator file system area has not been initialized. Enter the hdsfmkfs command to initialize it.

aa...aa: Name of the Datareplicator file system area

S: Cancels processing.

O: Initialize the file system area with the hdsfmkfs command, and then re-execute.

KFRB04806-E

Specified file is not character device, file system area = *aa...aa*.
(S)

A specified Datareplicator file system area is not a character special file.

aa...aa: Name of the Datareplicator file system area

S: Cancels processing.

O: Specify the correct character special filename and re-execute.

KFRB04807-E

Specified file is already used by another user, *aa...aa* = *bb...bb*,
file system area = *cc...cc*. (S)

A specified Datareplicator file system area is being used by another Datareplicator.

aa...aa: HDEPATH for the source system or HDSPATH for the target system

bb...bb: Name of the Datareplicator directory in use

cc...cc: Name of the Datareplicator file system area

S: Cancels processing.

O: Re-execute after the Datareplicator using the area has terminated.

KFRB04808-E

Invalid -l operand, shortage number of sectors, device sector =
aa...aa, minimum need sector = *bb...bb*. (S)

Operand (-l) specification is invalid. There are not enough sectors for the specified number of files.

aa...aa: Number of sectors in the Datareplicator file system area

bb...bb: Minimum number of sectors required for the specified files

S: Cancels processing.

O: Obtain the correct number of system files to be specified in the operand (-l), and then re-execute.

KFRB04809-Q

This area is already used by *aa...aa* = *bb...bb*.

Initialization for file system area, do you want to continue
initialization (y or n)?. (S)

Ask if you want to initialize the Datareplicator file system area. To initialize it, enter y; otherwise, enter n.

This message is displayed only when a Datareplicator file system area used by Datareplicator is to be initialized.

aa...aa: HDEPATH for the source system or HDSPATH for the target system

bb...bb: Name of the Datareplicator directory

S: Waits for the user's reply.

O: To initialize the Datareplicator file system area, enter y; otherwise, enter n.

KFRB04810-E

Current path get error occurred , errno = *aa...aa*. (S)

Acquisition of current path (`getcwd`) resulted in an error.

aa...aa: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute. Or, specify the absolute pathname of the Datareplicator file system area.

KFRB04811-E

I/O error occurred , type = *aa...aa*, file system area = *bb...bb*, errno = *cc...cc*. (S)

An input/output error occurred in a Datareplicator file system area.

aa...aa: Input or Output

bb...bb: Name of the Datareplicator file system area resulting in the error

cc...cc: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04812-E

Link I/O error occurred , type = *aa...aa*, file system area = *bb...bb*, link name = *cc...cc*, errno = *dd...dd*. (S)

A link input/output error occurred in a Datareplicator file system area.

aa...aa: Type of link

bb...bb: Name of the Datareplicator file system area

cc...cc: Link name

dd...dd: Error number set in `errno`

S: Cancels processing.

O: Eliminate the cause of the error and re-execute.

KFRB04813-E

Insufficient memory , size = *aa...aa*, errno = *bb...bb*, table = *cc...cc*. (S)

A memory shortage occurred.

aa...aa: Size of memory to be allocated

bb...bb: Error number set in `errno`

cc...cc: Name of table to be created in memory

S: Cancels processing.

O: Resolve the memory shortage and re-execute.

KFRB04814-E

Duplicate file name, system file name = *aa...aa*. (S)

A system file is duplicated.

aa...aa: System filename

S: Cancels processing.

O: Specify the correct system filename and re-execute.

KFRB04815-E

File number over flow, file system area = *aa...aa*, max file number = *bb...bb*. (S)

The number of files exceeded the maximum specified with the `hdsfmkfs` command.

aa...aa: Name of the Datareplicator file system area

bb...bb: Maximum number of files

S: Cancels processing.

O: Use the `hdsfstatfs` command to check the status of the file system area, increase the number of system files in the `hdsfmkfs` command or allocate the system file in another file system area, and then re-execute.

KFRB04816-E

Shortage of file size, file system area = *aa...aa*, area size = *bb...bb*. (S)

There is not enough space in a Datareplicator file system area.

aa...aa: Name of the Datareplicator file system area

bb...bb: Size of the Datareplicator file system area

S: Cancels processing.

O: Use the `hdsfstatfs` command to check the status of the file system area, allocate the system file in another file system area, and then re-execute.

KFRB04817-E

Too small sector size, file system area = *aa...aa*, sector size = *bb...bb*. (S)

The sector length is too small for a character special file. The sector length must be at least 256 bytes.

aa...aa: Name of the Datareplicator file system area

bb...bb: Sector length

S: Cancels processing.

O: Increase the sector length, and then re-execute.

KFRB04818-E

Exist specified file as character special file, file system area = *aa...aa*, storage file = *bb...bb*. (S)

A file to be stored in the Datareplicator file system area already exists as a character special file.

aa...aa: Name of the Datareplicator file system area

bb...bb: Name of the file to be stored

S: Cancels processing.

O: Delete or rename the file to be stored, and then re-execute.

KFRB04819-E

Invalid *-q* operand. (S)

Operand (*-q*) specification is invalid.

S: Cancels processing.

O: Check the operand (*-q*) specification and re-execute.

KFRB04820-E

Invalid *-n* operand. (S)

Operand (*-n*) specification is invalid.

S: Cancels processing.

O: Check the operand (*-n*) specification and re-execute.

KFRB04821-W

Ignore *aa...a* request, reason = *bb...b*, kind = *cc...c*, inf = *dd...d*. (C)

Request *aa...a* was ignored for the reason *bb...b*.

aa...a: Reset request:

send_counter_reset: Resetting of the data transmission count

reflect_counter_reset: Resetting of the import processing count

bb...b: Reason for ignoring the reset request:

process_starting: Process is starting

cc...c: Type of identifier:

send_id: Destination identifier

ds_id: Data linkage identifier

dd...d: Identifier

S: Resumes processing.

O: To reset the data-transmission count, do one of the following:

- Terminate the transmission process at the source, and then restart it with the `-r` option specified.
- If the `eventcntreset` parameter is specified in the transmission environment definition, issue the corresponding event.

To reset the import processing count, do one of the following:

- Terminate the source Datareplicator, and then restart it with the `-r` option specified.
- If the `eventcntreset` parameter is specified in the import environment definition, issue the corresponding event.

KFRB04822-I

It reset the number of the *aa...a*, kind = *bb...b*, inf = *cc...c*. (C)
aa...a was reset.

aa...a: Item that was reset:

send_counter: Data transmission count

reflect_counter: Import processing count

bb...b: Type of identifier:

send_id: Destination identifier

ds_id: Data linkage identifier

cc...c: Identifier

S: Resumes processing.

KFRB04901-E

Definition format error, file = *aa...aa*, line = *bb...bb*. (S)

The format of a conversion definition file is invalid.

aa...aa: Name of the conversion definition file

bb...bb: Line number

S: Cancels processing.

O: Correct the information on the indicated line in the conversion definition file, and then re-execute.

KFRB04902-E

Invalid character code, file = *aa...aa*, line = *bb...bb*, kind = *cc...cc*,
reason = *dd...dd*. (S)

A character code is invalid.

aa...aa: Name of the conversion definition file

bb...bb: Line number

cc...cc: Type:

before: Source character code

after: Target character code

dd...dd: Reason for the error

S: Cancels processing.

O: Correct the character code on the indicated line in the conversion definition file as appropriate to the indicated reason, and then re-execute.

KFRB04951-E

Command(*aa...aa*) execution error, code = *bb...bb*, info = *cc...cc*. (S)

An error occurred in the current file copy command.

aa...aa: Command name

Source: `hdefcopy`

Target: `hdsfcopy`

bb...bb: Error code

cc...cc: Detail information for each error code (if there is no output information, ** is displayed)

S: Cancels processing.

O: Take the following action according to the error code:

Error code	Details	Action
1	There is no duplexing control file.	File duplexing is not being used. Re-execute the command at a node where file duplexing is used.
2	The physical file at the copy source is not duplexed.	Specify a physical file that is used for duplexing and re-execute the command.
3	Two physical files do not constitute the same logical file.	Specify physical file names that constitute the same logical file and re-execute the command.

Error code	Details	Action
4	Two physical files have different file types.	Specify physical files whose file types are the same and re-execute the command.
5	The status of the source file is invalid.	Use the status display command to make sure that the status of the source file is <code>ACTIVE</code> . To copy the current file, specify the name of a source file that is in <code>ACTIVE</code> status and re-execute the command.
6	The status of the target file is invalid.	Use the status display command to make sure that the status of the target file is <code>HOLD</code> . To copy the current file, specify the name of a target file that is in <code>HOLD</code> status and re-execute the command.
7	Mapped file is invalid.	After restarting Datareplicator, re-execute the command.
11	Two physical files have different sector lengths. Detail information: Sector length of the source file	Specify physical files with the same sector length and re-execute the command.
12	The source file is invalid.	Either initialize Datareplicator or specify the correct file, and then re-execute the command.
13	The size of the source file is invalid.	Either initialize Datareplicator or specify the correct file, and then re-execute the command.
14	The size of the target file is insufficient. Detail information: Size of the source file.	Use a file that can store the amount of data indicated as detail information as the physical file and re-execute the command.
15	The specified file size exceeded the maximum value. Detail information: file name, detail code	See the action for <code>KFRB00069-E</code> .

KFRB04953-E

`Command(aa...aa)` execution error, code = `bb...bb`. (S)

An error occurred in the duplexing file status displays command.

aa...aa: Command name

Source: `hdefstate`

Target: `hdsfstate`

bb...bb: Error code

S: Cancels processing.

O: Take the following action according to the error code:

Error code	Details	Action
1	There is no duplexing control file.	File duplexing is not being used. Re-execute the command at a node where file duplexing is used.
2	The specified file is not duplexed.	Specify a physical file that is used for duplexing and re-execute the command.
3	The file is in unused status.	The specified file has never been used by Datareplicator. Specify the correct file name and re-execute the command.

KFRB04954-I

Command(*aa...aa*) execution success. (S)

Command execution was completed.

aa...aa: Command name

S: Resumes processing.

KFRB05001-I

Capture process started. (E)

Datareplicator is starting the extraction process.

S: Resumes processing.

KFRB05002-I

Capture process terminated. (E)

Datareplicator is terminating the extraction process.

S: Terminates processing.

KFRB05003-E

Capture process terminated abnormally, module = *aa...aa*, line = *bb...bb*. (E)

The extraction process is terminating abnormally.

aa...aa: Module name (internal information)

bb...bb: Line number (internal information)

S: Cancels processing.

O: Obtain a core dump and a shared memory dump, and then contact the customer support center.

KFRB05004-E

Internal error occurred in capture process, addinfo = [*aa...aa*], runid = *xx...xx*, blkno = *yy...yy*, recno = *zz...zz*. (E)

Datareplicator detected an internal conflict in the extraction process. Detail

information = [aa...aa], Run ID = xx...xx, Block number = yy...yy, Record number = zz...zz.

aa...aa: Error number

xx...xx: Run ID

yy...yy: System log block number

zz...zz: System log record number

S: Cancels processing.

O: See *10.4 List of cause codes* and take action appropriate to the displayed error number.

KFRB05005-E

This data-type is unable to be extracted, data-type = aa...aa. (E)

Datareplicator cannot extract this data type.

aa...aa: Data type

S: Cancels processing.

O: Obtain a shared memory dump, HiRDB system log file, system log file, and data linkage file, and then contact the customer support center.

KFRB05006-E

Error occurred in analyzing HiRDB log file, addinfo = [aa...aa, bb...bb], runid = xx...xx, blkno = yy...yy, recno = zz...zz. (E)

An error occurred during the analyzing of the HiRDB system log file. Detail information = [aa...aa, bb...bb], Run ID = xx...xx, Block number = yy...yy, Record number = zz...zz.

aa...aa: Detail information 1 (internal information)

bb...bb: Detail information 2 (internal information)

xx...xx: Run ID

yy...yy: System log block number

zz...zz: System log record number

S: Cancels processing. Datareplicator outputs internal information as the detail information.

O: Obtain a shared memory dump, HiRDB system log file, system log file, and data linkage file, and then contact the customer support center.

KFRB05007-E

Error occurred in reading HiRDB log file, addinfo = [aa...aa], runid = xx...xx, blkno = yy...yy, recno = zz...zz. (E)

An input error occurred on the HiRDB system log file. Detail information = [aa...aa],
Run ID = xx...xx, Block number = yy...yy, Record number = zz...zz.

aa...aa: Error number

xx...xx: Run ID

yy...yy: System log block number

zz...zz: System log record number

S: Cancels processing.

O: See *10.4 List of cause codes* and take an action appropriate to the displayed error number.

KFRB05008-I

Accepted stop request, request kind = aa...aa. (E)

Datereplicator accepted a termination request for the extraction process. Request type = aa...aa

aa...aa: Request type:

Normal: Normal termination request

S: Starts termination processing.

KFRB05009-E

Invalid version, HiRDB Datereplicator not compatible with HiRDB.
(E)

HiRDB and the HiRDB Datereplicator do not have matching versions.

S: Cancels processing.

O:

When the data linkage recovery facility is not used:

Use systems with matching versions.

When the data linkage recovery facility is used:

All errors specific to the data linkage recovery facility are output to the error information file for the KFRB05009-E message. Identify the nature of the error from the information displayed for `function:` in the message text. For details, see *9.6.5(2) Handling errors during data linkage recovery via unload log files*.

KFRB05010-E

Unable to access communication file due to permanent lock error.
(E)

Datereplicator was unable to access the data linkage file because lock has not been

released.

S: Cancels processing.

O: Obtain a shared memory dump, system log file, and data linkage file, and then contact the customer support center.

KFRB05011-E

HiRDB overwrites unextracted system-log. (E)

Overwriting occurred on the HiRDB system log subject to extraction processing.

S: Cancels processing.

O: To restart data linkage, synchronize the data linkage environments at the source and target, initialize them, and then re-create the target database on the basis of the source database before restarting HiRDB Datareplicator linkage.

KFRB05012-I

Replication-start-log of HiRDB captured, runid = xx...xx, blkno = yy...yy, recno = zz...zz. (E)

Datareplicator detected HiRDB's data linkage start log (HiRDB Datareplicator linkage start log).

xx...xx: Run ID

yy...yy: System log block number

zz...zz: System log record number

S: Resumes processing.

KFRB05013-I

Replication-end-log of HiRDB captured, runid = xx...xx, blkno = yy...yy, recno = zz...zz. (E)

Datareplicator detected HiRDB's data linkage termination log (HiRDB Datareplicator linkage termination log).

xx...xx: Run ID

yy...yy: System log block number

zz...zz: System log record number

S: Cancels processing.

O: To restart data linkage, synchronize the data linkage environments at the source and target, initialize them, and then re-create the target database on the basis of the source database before restarting HiRDB Datareplicator linkage.

KFRB05014-E

Unable to read HiRDB system-log, because the value of logiosize operand is smaller than pd_log_max_data_size operand of HiRDB, logiosize = aa...aa. (E)

Datareplicator cannot read the HiRDB system log because the logiosize operand value in the extraction environment definition file is smaller than the HiRDB's pd_log_max_data_size operand value.

aa...aa: Value of logiosize

S: Cancels processing.

O: Correct the logiosize operand value in the extraction environment definition file, and then restart.

KFRB05015-I

Normal termination process started, found normal termination log of HiRDB, runid = xx...xx, blkno = yy...yy, recno = zz...zz. (E)

Normal termination of the extraction process has started because Datareplicator detected HiRDB's normal termination log (EOF). Run ID = xx...xx, Block number = yy...yy, Record number = zz...zz.

xx...xx: Run ID

yy...yy: System log block number

zz...zz: System log record number

S: Terminates processing.

KFRB05016-E

HiRDB replication function is terminated. (E)

The HiRDB Datareplicator linkage facility has been terminated or cancelled.

S: Cancels processing.

O: To restart data linkage, synchronize the data linkage environments at the source and target, initialize them, and then re-create the target database on the basis of the source database before restarting HiRDB Datareplicator linkage.

KFRB05017-I

Table to be extracted is not defined in this server, server name = aa...aa. (E)

The table subject to extraction processing is not defined at this server.

aa...aa: Server name

S: Terminates processing.

KFRB05018-I

Found the end of HiRDB system-log, runid = *xx...xx*, blkno = *yy...yy*,
 recno = *zz...zz*. (E)

The end of the HiRDB system log was detected. Run ID = *xx...xx*, Block number =
yy...yy, Record number = *zz...zz*.

xx...xx: Run ID

yy...yy: System log block number

zz...zz: System log record number

S: Resumes processing.

KFRB05019-E

Read point in HiRDB system-log skipped, runid = *xx...xx*, blkno =
yy...yy, recno = *zz...zz*. (E)

The read pointer for the HiRDB system log was skipped. Run ID = *xx...xx*, Block
 number = *yy...yy*, Record number = *zz...zz*.

pdrplstop and *pdrplstart* might have been executed before HiRDB's extraction
 process finished extraction processing.

xx...xx: Run ID

yy...yy: System log block number

zz...zz: System log record number

S: Cancels processing.

O: To restart data linkage, synchronize the data linkage environments at the source and
 target, initialize them, and then re-create the target database on the basis of the source
 database before restarting HiRDB Datareplicator linkage.

KFRB05020-I

The value of *extsuppress* operand is true, extraction suppressed
 in this server, server name = *aa...aa*. (E)

Extraction processing is suppressed at this server.

aa...aa: Server name

S: Resumes processing.

KFRB05021-W

Ignored event-command because Event-table is not FIX-TABLE,
 runid = *xx...xx*, blkno = *yy...yy*, recno = *zz...zz*. (E)

Datareplicator ignored the event because the event control table is not a FIX-attribute
 table. Run ID = *xx...xx*, Block number = *yy...yy*, Record number = *zz...zz*.

xx...xx: Run ID

yy...yy: System log block number

zz...zz: System log record number

S: Resumes processing. Note that events are not usable because no event information is being collected.

O: Change the operation mode so that events will not be used.

[Action]

After completion of extraction processing, terminate the source Datareplicator, re-create the event control table, and then re-execute the hdeprep command. Then restart extraction processing.

KFRB05022-W

EOF block is not the last block of read blocks, runid = xx...xx, blkno = yy...yy, recno = zz...zz. (E)

The last block read was not the EOF block. Run ID = xx...xx, Block number = yy...yy, Record number = zz...zz.

Datareplicator issues this message when it detects a HiRDB system log that cannot be extracted with this data linkage.

xx...xx: Run ID

yy...yy: System log block number

zz...zz: System log record number

S: Resumes processing.

KFRB05023-W

Ignored column-log, runid = xx...xx, blkno = yy...yy, recno = zz...zz. (E)

Datareplicator ignored the column-based log. Run ID = xx...xx, Block number = yy...yy, Record number = zz...zz.

Datareplicator issues this message when it detects a column-based HiRDB system log that is not subject to data linkage.

xx...xx: Run ID

yy...yy: System log block number

zz...zz: System log record number

S: Resumes processing.

KFRB05025-E

Definition inconsistent. the value of extsuppress operand is true, but table of this server is defined in extract definition, server name = aa...aa. (E)

Extraction has been suppressed at this server, but this server's table is specified in the extraction definition as being subject to extraction processing.

aa...aa: Server name

S: Cancels processing.

O: To restart data linkage, synchronize the data linkage environments at the source and target, initialize them, and then re-create the target database on the basis of the source database before restarting HiRDB Datareplicator linkage.

KFRB05026-I

Normal termination process started, found planned termination log of HiRDB, runid = *xx...xx*, blkno = *yy...yy*, recno = *zz...zz*. (E)

Normal termination of the extraction process has started because Datareplicator detected HiRDB's planned termination log (EOF). Run ID = *xx...xx*, Block number = *yy...yy*, Record number = *zz...zz*.

xx...xx: Run ID

yy...yy: System log block number

zz...zz: System log record number

S: Terminates processing.

KFRB05027-W

Ignored event-command because specified send identifier is wrong, event code = *aa...aa*, send identifier = *bb...bb*. (E)

Datareplicator ignored an event because the specified target identifier was invalid.

aa...aa: Event code

bb...bb: Target identifier

S: Ignores the event and resumes processing.

KFRB05028-E

Exception data were detected, exception data = *aa...aa*, message-id = *bb...bb*. (E)

Datareplicator detected exception data.

aa...aa: Exception data

bb...bb: Message ID in the advanced queue

S: Terminates processing.

O: Specify data replacement so that data linkage is supported, and then restart extraction processing.

KFRB05029-W

Exception data were detected, exception data = *aa...aa*, extraction data type = *bb...bb*, substitution type = *cc...cc*, message-id = *dd...dd*.
(E)

Datareplicator detected exception data.

aa...aa: Exception data

bb...bb: Extraction data type

cc...cc: Replacement pattern:

 null: Replace with the null value

 high: Replace with the maximum value

 low: Replace with the minimum value

dd...dd: Message ID in the advanced queue

S: Resumes processing.

KFRB05030-E

Extract data does not belong to extraction target were detected.
table-id = *aa...aa*. (E)

Datareplicator detected update information that does not belong to a table that is subject to extraction processing.

aa...aa: Table ID

S: Terminates processing.

O: To restart data linkage, you must reset it.

KFRB05031-W

The number of vacant transmission queue files decreased to *dd...dd*.
(C)

The number of available extraction information queue files has decreased to *dd...dd*.

dd...dd: Number of extraction information queue files

S: Resumes processing.

O: Check to see if transmission processing has started. If transmission processing has not started, start it.

KFRB05034-I

Queue file recovery started, first file = *aa...aa*, last file = *bb...bb*
(C)

Recovery of the extraction information queue file has started.

aa...aa: First file name in the extraction information queue file to be recovered

(qufile001 to qufile016).

bb...bb: Last file name in the extraction information queue file to be recovered (qufile001 to qufile016).

S: Resumes processing.

KFRB05035-I

Queue file recovery completed, first file = *aa...aa*, last file = *bb...bb* (C)

Recovery of the extraction information queue file has been completed.

aa...aa: First file name in the extraction information queue file that has been recovered (qufile001 to qufile016).

bb...bb: Last file name in the extraction information queue file that has been recovered (qufile001 to qufile016).

S: Resumes processing.

O: Check that the range of files of recovered matches the information displayed in the KFRB05034-I message that was output when recovery started.

KFRB05037-W

Queue file that was formatted in an old version exists. (C)

The facility for recovering the extraction information queue file cannot be used because there is an extraction information queue file that was initialized by an older version of HiRDB Datareplicator.

S: Resumes processing.

O: If the facility for recovering the extraction information queue file is used in the system, execute an initial start on the source Datareplicator. If the facility for recovering the extraction information queue file is not used in the system, there is no need to execute an initial start.

KFRB05038-W

All of queue files are in an initial state. (C)

There is no need to execute the facility for recovering the extraction information queue file because all extraction information queue files are in initial status.

S: Stops processing.

KFRB05039-E

Unable to execute queue file recovery function, reason = *aa...aa*. (C)

A condition for using the facility for recovering the extraction information queue file is not satisfied.

aa...aa: Reason code

syncterm:

The facility for recovering the extraction information queue file cannot be used when `true` is specified in the `syncterm` operand in the extraction system definition.

S: Stops processing.

O: Use the data linkage recovery facility to perform the recovery.

KFRB05040-I

Datareplicator recovery function started. (C)

Recovery using the data linkage recovery facility (system log method) has started.

S: Resumes processing.

KFRB05041-I

Skip of update information for Datareplicator recovery function completed, additional transaction info = *aa...aa*. (C)

Recovery by the data linkage recovery facility has been completed. Datareplicator will now resume normal data linkage processing and sending update information.

aa...aa: Extraction transaction information in the last update information that was skipped

S: Resumes processing.

KFRB05042-E

Recover information file is invalid, file name = *aa...aa*, reason = *bb...bb*. (C)

The recovery information file is invalid.

aa...aa: File name

bb...bb: Reason code

file not found:

There is no recovery information file under the directory specified in the `HDEPATH` environment variable.

invalid file format:

The format of the recovery information file is invalid.

invalid recover information condition:

Because the recovery information in the recovery information file has not been synchronized, recovery cannot be performed using this recovery information.

S: Cancels processing.

O: Take appropriate action according to the following table.

Reason code	Description	Action
file not found	There is no recovery information file under the directory specified in the HDEPATH environment variable.	Make sure that the following are true: <ul style="list-style-type: none"> The recovery information file has been transferred under the directory specified in the HDEPATH environment variable. The name of the recovery information file is correct.
invalid file format	The format of the recovery information file is invalid.	Make sure that the following are true: <ul style="list-style-type: none"> The recovery information file transfer method is binary mode. If the recovery information file is a character special file, the correct <code>count</code> value was specified in the <code>dd</code> command when the file was copied to a regular file. If neither of the above applies, the recovery information file has become corrupted. Execute data linkage recovery by using unload log files.
invalid recover information condition	Because the recovery information in the recovery information file has not been synchronized, recovery cannot be performed by using this recovery information.	Execute data linkage recovery by using unload log files.

KFRB06001-I

Agent process started. (L)

The agent has started.

S: Resumes processing.

KFRB06002-I

Agent process terminated. (L)

The agent was terminated.

S: Terminates processing.

KFRB06003-I

Variable of agent option is changed. (L)

Agent settings were changed.

S: Resumes processing.

KFRB06004-E

Agent process is already started. (S+L)

The agent is already active.

S: Cancels startup processing.

KFRB06005-E

Agent process is not started. (S+L)

The agent is not active.

S: Cancels termination processing.

KFRB06006-E

Agent process start failed. (S)

Agent startup processing resulted in an error.

S: Cancels startup processing.

O: Check the error message issued immediately before and eliminate the cause of the error, and then re-execute.

KFRB06007-W

Collector process's output is invalid, process name = *aa...aa*. (L)

Output of the information collection process is invalid.

aa...aa: Process name

S: Attempts the processing for up to five times at the specified monitoring interval, and then terminates processing if the error has not been eliminated.

O: Datareplicator might not be installed correctly. If so, reinstall Datareplicator and restart it. If the same warning is issued again, contact the customer support center.

KFRB06008-W

Collector process error occurred, process name = *aa...aa*, exit code = *bb...bb*. (L)

The information collection process resulted in an error.

aa...aa: Process name

bb...bb: Process termination code

S: Attempts the processing for up to five times at the specified monitoring interval, and then terminates processing if the error has not been eliminated.

O: Eliminate the cause of the error.

KFRB06009-W

Cm2 systemtrap command error occurred, info = *aa...aa*. (L)

An error occurred in JP1/Cm2's systemtrap command.

aa...aa: systemtrap command's error message

If the message exceeds 255 bytes, the systemtrap command outputs only the first 255 bytes of the message.

S: Attempts the processing for up to five times at the specified monitoring interval, and then terminates processing if the error has not been eliminated.

O: Eliminate the cause of the error associated with JP1/Cm2's `systemtrap` command. If the `$OV_BIN` environment variable setting is wrong, correct it, and then restart.

KFRB06010-E

Other process is changing agent option. (S+L)

Another process is updating the agent settings.

S: Cancels the processing to update agent settings.

O: Wait a while, and then re-execute.

KFRB06011-I

Command request was accepted, command = *aa...aa*. (L)

A command request was accepted.

aa...aa: Command line

S: Resumes processing.

KFRB06012-E

File access error was occurred, operation = *aa...aa*, file = *bb...bb*,
errno = *cc...cc*. (S+L)

A file manipulation error was detected.

aa...aa: Manipulation type

bb...bb: Filename

cc...cc: Error number set in errno

Error number 0 indicates the following:

read: Invalid file contents

write: Insufficient disk space

S: Cancels processing.

O: Check `errno.h` or the user's OS documentation for the error number and eliminate the cause of the error.

KFRB06013-E

Failed to connect service control manager, code = *aa...aa*, errno =
bb...bb. (S+L)

Connection establishment with the service control manager failed.

aa...aa: Code

bb...bb: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Check `errno.h` or the user's OS documentation for the error number and eliminate the cause of the error.

KFRB06014-E

Failed to set state of event object to signaled, function name = *aa...aa*, event name = *bb...bb*, errno = *cc...cc*. (S+L)

Event setup failed.

aa...aa: Function name

bb...bb: Event name

cc...cc: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Check `errno.h` or the user's OS documentation for the error number and eliminate the cause of the error.

KFRB06015-E

Invalid setup information. (S+L)

Setup information is invalid.

S: Cancels processing.

O: Reinstall Datareplicator.

KFRB06016-E

Failed to regist service handler function, errno = *aa...aa*. (L)

Registration of the service control handler failed.

aa...aa: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Check `errno.h` or the user's OS documentation for the error number and eliminate the cause of the error.

KFRB06017-E

Failed to create event object, function name = *aa...aa*, event name = *bb...bb*, errno = *cc...cc*. (L)

Creation of an event object failed.

aa...aa: Function name

bb...bb: Event name

cc...cc: Error number obtained by `GetLastError()`

S: Cancels processing.

O: Check `errno.h` or the user's OS documentation for the error number and eliminate the cause of the error.

KFRB06018-E

Agent process terminated abnormally, error code = *aa...aa*. (L)

The agent terminated abnormally.

aa...aa: Error code

S: Cancels processing.

O: Check the manual for the message corresponding to the error code and eliminate the cause of the error.

KFRB06101-E

Signal operation failure, process-id = *aa...aa*, kind = *bb...bb*, errno = *cc...cc*. (S+L)

Signal operation failed.

aa...aa: Remote process ID

bb...bb: Signal to be sent

cc...cc: Error number set in `errno`

S: Cancels processing.

O: Check `errno.h` or the user's OS documentation for the error number and eliminate the cause of the error.

KFRB06102-E

Cannot exec child process, process name = *aa...aa*, errno = *bb...bb*. (S+L)

Datareplicator was unable to execute a subprocess.

aa...aa: Subprocess name

bb...bb: Error number set in `errno`

S: Cancels processing.

O: Check `errno.h` or the user's OS documentation for the error number and eliminate the cause of the error.

KFRB06103-E

Cannot fork child process, errno = *aa...aa*. (S+L)

Datareplicator was unable to create a subprocess.

aa...aa: Error number set in `errno`

S: Cancels processing.

O: Check `errno.h` or the user's OS documentation for the error number and eliminate the cause of the error.

KFRB06104-E

Invalid command argument. (S+L)

A specified command argument is invalid.

S: Cancels processing.

O: Check the manual for the correct command arguments and re-execute.

KFRB06105-E

Option value is invalid, option name = *aa...aa*. (S+L)

The value of a command option is invalid.

aa...aa: Command option

S: Cancels processing.

O: Check the manual for details, such as the permitted value range for the command option, and then re-execute.

KFRB06106-E

Insufficient memory, type = *aa...aa*, size = *bb...bb*, errno = *cc...cc*. (L)

Datereplicator was unable to allocate memory.

aa...aa: Detail code

bb...bb: Memory allocation size

cc...cc: Error number set in `errno`

S: Cancels processing.

O: Check `errno.h` or the user's OS documentation for the error number and eliminate the cause of the error.

KFRB06201-E

Install directory not found. (L)

Datereplicator was unable to find the Datereplicator installation directory. Registry information is wrong.

S: Cancels processing.

O: Reinstall Datereplicator.

KFRB06202-E

Insufficient memory for SNMP request, size = *aa...aa*, errno = *bb...bb*. (L)

At the SNMP agent, Datereplicator was unable to allocate sufficient memory to execute the information collection or remote control requested by the supervisor.

aa...aa: Memory allocation size

bb...bb: Error number set in `errno`

S: Cancels the requested information collection or remote control operation.

O: Check the user's OS documentation for the error number and eliminate the cause of the error. Then re-execute information collection or remote control at the supervisor.

KFRB06203-E

Insufficient memory for MIB analyze, size = *aa...aa*, `errno` = *bb...bb*.
(L)

Datereplicator was unable to allocate sufficient memory to analyze the MIB file at the SNMP agent.

aa...aa: Memory allocation size

bb...bb: Error number set in `errno`

S: Terminates the SNMP agent.

O: Check the user's OS documentation for the error number and eliminate the cause of the error. Then restart the SNMP service.

KFRB06204-E

Invalid command in MIB file, object id = *aa...aa*. (L)

An invalid command was detected during MIB file analysis at the SNMP agent. The MIB file might be damaged.

aa...aa: Object ID specifying the invalid command

S: Terminates the SNMP agent.

O: Reinstall Datereplicator, recover the MIB file, and then restart the SNMP service.

KFRB06205-E

Collector process time over for 10 sec, process = *aa...aa*. (L)

The information collection process could not collect information within 10 seconds at the SNMP agent.

aa...aa: Detail information about the information collection process

S: Cancels the information collection process.

O: Take an appropriate action on the basis of the message output by the information collection process, and then re-execute information collection from the supervisor.

KFRB06206-E

Collector process error occurred, process = *aa...aa*. (L)

An error occurred in the information collection process at the SNMP agent.

aa...aa: Detail information about the information collection process

S: Terminates the information collection process.

O: Take an appropriate action on the basis of the message output by the information collection process, and then re-execute information collection from the supervisor.

KFRB06207-E

Invalid SNMP request type, request type = *aa...aa*. (L)

The attribute of the object ID requested for information collection or remote control was invalid at the SNMP agent. The SNMP agent can process only the `INTEGER`, `Counter`, and `DisplayString` attributes.

aa...aa: Attribute code of the object ID

S: Terminates the requested information collection or remote control operation.

O: Check the attribute of the object ID requested by the supervisor, and then re-execute information collection or remote control from the supervisor.

KFRB06208-E

Invalid SNMP request code, request code = *aa...aa*. (L)

The request code for information collection or remote control is invalid at the SNMP agent. The SNMP agent can process the `GET`, `GETNEXT`, and `SET` request codes.

aa...aa: Request code

S: Terminates the requested information collection or remote control operation.

O: Check the supervisor's request code and re-execute information collection or remote control from the supervisor.

KFRB06209-E

Invalid SNMP specified value size, value size = *aa...aa*. (L)

The length of the character string specified for a remote control parameter is invalid at the SNMP agent.

aa...aa: Length of character string specified for the parameter

S: Terminates the requested remote control operation.

O: Check the length of character string specified for the remote control parameter, and then re-execute remote control operation from the supervisor.

KFRB06210-E

Invalid SNMP specified value, value = *aa...aa*. (L)

A numeric parameter specified for remote control is invalid at the SNMP agent.

aa...aa: Specified numeric value

S: Terminates the requested remote control operation.

O: Check the maximum and minimum values for the remote control parameter, and

then re-execute remote control from the supervisor.

KFRB06211-E

Process create error occurred, process = *aa...aa*, error code = *bb...bb*. (L)

Datareplicator cannot create the information collection process at the SNMP agent for information collection or remote control.

aa...aa: Process information to be created

bb...bb: Error number returned by the `CreateProcess` function

S: Cancels information collection or remote control operation.

O: Check the user's OS documentation for the error number, eliminate the cause of the error, and then re-execute information collection or remote control from the supervisor.

KFRB06212-E

Wait for process failed, process = *aa...aa*, error code = *bb...bb*. (L)

Datareplicator was unable to wait for completion of the information collection process at the SNMP agent.

aa...aa: Process information to be completed

bb...bb: Error number returned by the `WaitForSingleObject` function

S: Cancels information collection.

O: Check the user's OS documentation for the error number, eliminate the cause of the error, and then re-execute information collection or remote control from the supervisor.

KFRB06213-E

Unable to get a process exit code, process = *aa...aa*, error code = *bb...bb*. (L)

Datareplicator was unable to obtain the termination code of the information collection process at the SNMP agent.

aa...aa: Process information whose termination code is to be obtained

bb...bb: Error number returned by the `GetExitCodeProcess` function

S: Cancels information collection.

O: Check the user's OS documentation for the error number, eliminate the cause of the error, and then re-execute information collection or remote control from the supervisor.

KFRB06501-E

Specified DSN name is not for SQL Server. (C + S)

An invalid DSN was specified. The specified DSN name is not for SQL Server.

S: Cancels processing.

O: An invalid DSN name was specified in one of the following:

- Extraction system definition
- hdeprepS -D
- hdeeventsS -D
- hderesstatesS -D

Specify the correct DSN name and re-execute the command.

KFRB06503-E

Error occurred in object creation, object type = *aa...aa*, object name = *bb...bb*, errcode = *cc...cc*, info = *dd...dd* (C + S)

Creation of an SQL Server object failed.

aa...aa: Object type

Index: Index

Table: Table

Trigger: Trigger

bb...bb: Object name

cc...cc: SQLSTATE

dd...dd: SQL Server error message

S: Cancels processing.

[Action]

Check the SQL Server manual, eliminate the cause of the error on the basis of the SQLSTATE error code and SQL Server error message, and then re-execute the command.

KFRB06504-E

Object does not exist, object type = *aa...aa*, object name = *bb...bb*, errcode = *cc...cc*, info = *dd...dd* (C + S)

The SQL Server object does not exist.

aa...aa: Object type

Table: Table

bb...bb: Object name

cc...cc: SQLSTATE

dd...dd: SQL Server error message

S: Cancels processing.

[Action]

Check the SQL Server manual, eliminate the cause of the error on the basis of the `SQLSTATE` error code and SQL Server error message, and then re-execute the command.

KFRB06505-E

SQL Server error, func = *aa...aa*, return = *bb...bb*, errcode = *cc...cc*, info = *dd...dd* (C + S)

An error occurred on SQL Server.

aa...aa: Function name

bb...bb: Return value

cc...cc: `SQLSTATE`

dd...dd: SQL Server error message

S: Cancels processing.

[Action]

Check the SQL Server manual, eliminate the cause of the error on the basis of the error code `SQLSTATE` and SQL Server error message, and then re-execute the command.

KFRB06506-E

SQL Server version error. This SQL Server version is not supported. (C + S)

The SQL Server version is not supported.

S: Cancels processing.

[Action]

Check the Datareplicator Extension manual and use a supported version of SQL Server.

KFRB06507-E

`QUEUE_ID` value in the queue table has exceeded *aa...aa* (99% of its maximum value). (C)

The value of `QUEUE_ID` for `RPL_EXT_QUEUE_TBL` exceeded *aa...aa* (99% of the maximum value).

aa...aa: 99% of the maximum value of `QUEUE_ID`

S: Cancels processing.

[Action]

Execute initial start of the source Datareplicator.

KFRB06508-W

"hderesstates" command may show previous/default value as resource operand table could not be updated with *aa...aa*, func = *bb...bb*, return = *cc...cc*, errcode = *dd...dd*, info = *ee...ee* (C)

Datereplicator was unable to update the value of RPL_EXT_RESOPDTBL to *aa...aa*.

aa...aa: Value in seconds that was obtained by converting the *sqls_msgkeep*time operand value in the extraction system definition

bb...bb: Function name

cc...cc: Return value

dd...dd: SQLSTATE

ee...ee: SQL Server error message

S: Resumes processing.

[Action]

Check the SQL Server manual, eliminate the cause of the error on the basis of the SQLSTATE error code and SQL Server error message, terminate and restart the source Datereplicator, and then re-execute the command.

KFRB09084-E

Specified group name not found, *aa...a*. (S+L)

A specified import group name was not found.

aa...a: Import group name

S: Cancels processing.

[Action]

Specify the correct group name in the command option.

KFRB09087-E

Output information exceeds the limits *aa...a*[KB]. (S+L)

Output of information has been cancelled because the size of the analysis results output file exceeds the maximum (-k option value).

aa...a: Maximum size of the analysis result output file (-k option value)

S: Cancels processing.

[Action]

Either increase the size specified in the -k option of the *hdsrefin*fm command or reduce the value specified in the -t option.

KFRB09088-W

The information at requested number can't be output, kind = *aa...a*, req-num = *bb...b*, out-num = *cc...c*, reason = *dd...d*. (S+L)

The command is unable to output as many transaction information items as requested (-t option value). The command outputs as many information items as it can, as indicated in *cc...c*.

aa...a: Type of transaction information:

un-reflect: Unimported transaction information

reflected: Imported transaction information

bb...b: Number of items requested (-t option value)

cc...c: Number of items that can be output

dd...d: Reason code:

no more data exist in queue file:

The import information queue file contains no more information than the count *cc...c*, or the import information queue file storing the information has already been overwritten.

definition change was detected:

Because a change to the definition was detected, older imported transaction information cannot be output.

maximum number:

The maximum number of items that can be output (2,000,000,000) was reached.

S: Resumes processing.

KFRB09089-E

Option value is invalid, Option = *aa*. (S+L)

An invalid value was specified in the command option.

aa: Command option

S: Cancels processing.

[Action]

Specify the correct value in the command option.

KFRB09090-W

Length of the character string exceed 45 bytes, resource = *aa...a*, id = *bb...b*. (S+L)

The length of an authorization identifier, table identifier, or column name exceeded 45

bytes. These names are not output to the `hdsrefinfmt` command's output result (only the table ID or column ID is displayed).

aa...a: Type of resource:

TableName: Authorization identifier or table identifier

ColumnName: Column name

bb...b: Table ID or column ID

S: Resumes processing.

KFRB09091-E

Invalid environment variable value, variable name = *aa...a*. (S+L)

An invalid value was specified in the environment variable *aa...a*.

aa...a: Name of environment variable:

HDS_RFI_ELANG: Character codes in the source database

HDS_RFI_PLANG: Character codes in the import system

S: Cancels processing.

[Action]

Set the correct value in the environment variable.

KFRB09092-E

The option requires an argument, Option = *aa*. (S+L)

Command option *aa* requires an option argument.

aa: Command option

S: Cancels processing.

[Action]

Specify the correct command option.

KFRB09093-E

System call error(function=fstat), Filename=*aa...a*, errno=*bb...b*.
(S+L)

The `fstat` system call returned an error.

aa...a: File name

bb...b: Error number

S: Cancels processing.

[Action]

Check the OS documentation for the error based on the error number, and then

eliminate the cause of the error.

KFRB09094-E

Internal error, *aa...a*. (S+L)

An internal conflict occurred.

aa...a: Maintenance information

S: Terminates processing.

[Action]

Contact the customer support center.

KFRB09200-E

Insufficient memory[*aa...aa*], require size = *bb...bb*. (S+L)

Memory allocation failed.

aa...aa: Type of memory to be allocated

bb...bb: Requested size

S: Cancels processing.

[Action]

Terminate unneeded processes or commands and re-execute.

KFRB09201-E

Duplicate option specified, Option name = *aa...aa*. (S)

A parameter (option) is duplicated.

aa...aa: Duplicated parameter (option) name

S: Cancels processing.

[Action]

Delete the duplicated parameter (option) and then re-execute.

KFRB09202-E

Exceeds max number of option, Option name = *aa...aa*. (S)

The number of specified parameters (options) exceeds the maximum.

aa...aa: Name of the parameter (option) that was specified too many times

S: Cancels processing.

[Action]

Reduce the number of parameters (options) and then re-execute.

KFRB09203-E

Unrecognized option specified. (S)

An invalid parameter (option) is specified.

S: Cancels processing.

[Action]

Delete the invalid parameter (option) and then re-execute.

KFRB09204-E

Necessary option not specified, Option name = *aa...aa*. (S)

A required parameter (option) is missing.

aa...aa: Parameter (option) name

S: Cancels processing.

[Action]

Specify the required parameter (option) and then re-execute.

KFRB09205-E

Specified value is out of range, Option name = *aa...aa*. (S)

A specified parameter (option) is outside the permitted range of values.

aa...aa: Parameter (option) name

S: Cancels processing.

[Action]

Correct the parameter (option) value, and then re-execute.

KFRB09206-E

Conflict options are specified, Option name = *aa...aa* and *bb...bb*.
(S)

Mutually exclusive parameters (options) are specified.

aa...aa, *bb...bb*: parameter (option) names

S: Cancels processing.

[Action]

Specify only one of the parameters (options) and then re-execute.

KFRB09207-W

Unrecognized value specified, ignore this. (S)

Datareplicator ignored a parameter (option) whose value was outside the permitted range of values.

S: Resumes processing; the results might not be as expected.

[Action]

Reevaluate the value of the parameter (option) after the next execution.

KFRB09208-E

Unanalyzable option specified, argc count=*aa...aa*, analyze parameter count=*bb...bb*.

Not all parameters (options) could be analyzed.

aa...aa: Number of specified parameters

bb...bb: Number of analyzed parameters

S: Cancels processing.

[Action]

Reevaluate the format of the specified parameters (options).

KFRB09209-E

Too long path name, Option name = *aa...aa*. (S)

The pathname of the file specified in a parameter (option) exceeds 255 bytes.

aa...aa: Parameter (option) name

S: Cancels processing.

[Action]

Check the specified parameter (option) value. If it exceeds 255 bytes, change the current directory.

KFRB09210-E

Specified value format error, Option name = *aa...aa*. (S)

The specification format of a parameter (option) is invalid.

aa...aa: Parameter (option) name

S: Cancels processing.

[Action]

Correct the specification format and re-execute.

KFRB09211-E

End-date is smaller than start-date, Option name = *aa...aa*. (S)

The end date precedes the start date.

aa...aa: Parameter (option) name

S: Cancels processing.

[Action]

Correct the date specification and re-execute.

KFRB09212-I

Usage: `hdstrcredit -f Input-file[, Input-file][-o Output-file][-l Level][-p pid[, -p pid ...]...][-t YYYYMMDDHHMMSS[, YYYYMMDDHHMMSS][-O Options]` (S)

This message lists the parameters (options) of the `hdstrcredit` command.

S: Cancels processing (this message is issued when an error is detected in one of the displayed parameters (options)).

KFRB09213-E

System call error (function=*aa...aa*), `errno = bb...bb`. (S+L)

A system call resulted in an error.

aa...aa: Name of the system call resulting in the error

bb...bb: Error number returned from the system call

S: Cancels processing.

[Action]

Check the cause of the error, eliminate it, and then re-execute.

KFRB09214-E

Specified file not found, file name = *aa...aa*. (S)

A specified file was not found.

aa...aa: Filename

S: Cancels processing.

[Action]

Correct the parameter (option) and re-execute.

KFRB09215-E

No access permission, file name = *aa...aa*. (S)

File access was denied (no permission).

aa...aa: Filename

S: Cancels processing.

[Action]

Obtain permission to access the file or have an authorized user access the file.

KFRB09216-E

No record in file, file name = *aa...aa*. (S)

File is empty.

aa...aa: Filename

S: Cancels processing.

[Action]

Specify the correct filename, and then re-execute.

KFRB09217-E

Invalid contents in file, file name = *aa...aa*. (S)

File format is invalid.

aa...aa: Filename

S: Cancels processing.

[Action]

Specify the correct filename, and then re-execute.

KFRB09301-E

Specified file not found, file-type = *aa...aa*, file name = *bb...bb*.
(S)

A specified file was not found.

aa...aa: File type

NMTSTATFILE: An invalid server name was specified in the HDEPATH
environment variable or in the -b option.

bb...bb: File name

S: Cancels processing.

O: Specify the correct value, and then re-execute the command.

KFRB09302-E

Unable to execute, because other command is not stopped. (S)

Datareplicator cannot execute this command because another command is executing.

S: Cancels processing.

O: Re-execute this command after the current command has terminated.

KFRB09303-E

Source site Datareplicator is not stopped. (S)

The source Datareplicator is running.

S: Cancels processing.

O: Terminate the source Datareplicator, and then re-execute the command.

KFRB09304-E

Specified queue file is in use, file name = *aa...aa*. (S)

Registration of the specified queue file cannot be released because the file is in use.

aa...aa: File name

S: Cancels processing.

O: Re-execute the command when the extraction information queue file is not being used.

KFRB09305-E

Transmission queue file cannot be added any more, because *nn...nn* queue files already exist. (S)

No more extraction information queue files can be registered because the maximum number of files have already been registered.

nn...nn: Number of extraction information queue files that have been registered

S: Cancels processing.

KFRB09306-E

Specified file is already in use, file-type = *aa...aa*, file name = *bb...bb*. (S)

The specified file has already been registered as an extraction information queue file.

aa...aa: File type

QUEUEFILE: Extraction information queue file

bb...bb: File name

S: Cancels processing.

O: Specify another file name.

KFRB09307-E

No more transmission queue files can be removed, because at least *nn...nn* queue files are required. (S)

No more extraction information queue file can be released from registration.

nn...nn: Number of extraction information queue files that are registered

S: Cancels processing.

KFRB09308-I

Transmission queue file was added successfully. (S)

Registration of an extraction information queue file was successful.

S: Terminates the command.

KFRB09309-I

Transmission queue file was removed successfully. (S)

Extraction information queue file registration release was successful.

S: Terminates the command.

KFRB09310-E

Specified file not found in source site system, file-type = *aa...aa*, file name = *bb...bb*. (S)

A specified file is not registered in the source system.

aa...aa: File type

EXTQUEUEFILE: Extraction information queue file name specified in the HDEPATH environment variable or in the -d option is invalid.

bb...bb: File name

S: Cancels processing.

O: Check and, if necessary, revise the specification, and then re-execute the command.

KFRB09311-E

Specified file was not found, the file must exist as a character special file, filename = *aa...aa*. (S)

A nonexistent file was specified in the -a option. The file specified in the -a option must be an existing character special file because the allocated queue file is either a character special file or stored in the Datareplicator file system area.

aa...aa: File name specified in the -a option

S: Cancels processing.

O: Create the character special file to be added as a queue file, and then re-execute the command.

KFRB09312-E

Dualing information for queue file was not found in the specified dual file definition, filename = *aa...aa*. (S)

The duplexing definition file specified in the -n option does not contain the duplexing definition for the queue file specified in the -a option.

aa...aa: File name specified in the -a option

S: Cancels processing.

O: Specify queue file information in the duplexing definition file for the queue file that is to be added, and then re-execute the command.

KFRB09313-E

Datareplicator is not use file duplicate function. (S)

Because Datareplicator is not using the file duplexing facility, you cannot add duplexed queue files.

S: Cancels processing.

O: Re-execute the command without duplexing the queue file to be added (without

specifying the `-n` option).

KFRB10001-I

The copy command of active-file was executed, file(*aa...aa*). (L)

The command that copies the current file executed successfully.

aa...aa: File name

S: Resumes processing.

KFRB10002-E

The mapped-file is broken, file(*aa...aa*).Please restart Datareplicator. (L)

The mapped file has been damaged.

aa...aa: Logical file name

S: Cancels startup processing.

O: Restart HiRDB Datareplicator.

KFRB10003-E

An active file does not exist.logicalfile =
aa...aa(A:*bb...bb*,B:*cc...cc*). (L)

There is no available current file.

aa...aa: Logical file name

bb...bb: Status of the primary file

cc...cc: Status of the secondary file

S: Cancels startup processing.

O: Use the data linkage recovery facility to recover, re-initialize, and then re-execute.

KFRB10004-I

A state of a file was changed. *aa...aa*->*bb...bb*(*cc...cc*). (L)

A file's status was changed.

aa...aa: Status before change

bb...bb: Status after change

cc...cc: File name

S: Resumes processing.

KFRB10005-I

A file was opened. logicalfile = *aa...aa*(A:*bb...bb*, B:*cc...cc*). (L)

A file was opened.

aa...aa: Logical file name

bb...bb: Status of the primary file
cc...cc: Status of the secondary file
 S: Resumes processing.

KFRB10006-W

A process is downing in writing file. logicalfile =
aa...aa(A:*bb...bb*, B:*cc...cc*). (L)

Process terminated while writing to a file.

aa...aa: Logical file name
bb...bb: Status of the primary file
cc...cc: Status of the secondary file
 S: Resumes processing.

KFRB10007-I

File copy from *aa...aa* to *bb...bb* completed. (S + L)

File copying was completed.

aa...aa: Source (normal file)
bb...bb: Target (erroneous file)
 S: Resumes processing.

KFRB10008-W

The sector length of a file is different, file(*aa...aa*, *bb...bb*),
 sector-size:*cc...cc*. (L)

The file has a different sector length.

aa...aa: Source (normal file)
bb...bb: Target (erroneous file)
cc...cc: Sector length
 S: Resumes processing.
 O: Set the target RAW device's sector length to the same value as for the source.

KFRB10009-W

The source copying file is invalid, file(*aa...aa*). (L)

The source file is invalid.

aa...aa: Source (normal file)
 S: Resumes processing.
 O: The source file is invalid. This message is displayed when no current file is

available. After using the data linkage recovery facility to recover the file, initialize it and re-execute the command.

KFRB10010-W

The file size of a copying file is invalid, file(*aa...aa*). (L)

The size of the source file is invalid.

aa...aa: Source (normal file)

S: Resumes processing.

O: The source file is invalid. This message is displayed when no current file is available. After using the data linkage recovery facility to recover the file, initialize it and re-execute the command.

KFRB10011-W

The disk capacity of a copy place is insufficient, file(*aa...aa*),
copy-size:*bb...bb*. (L)

There is not enough disk space at the target.

aa...aa: Target (erroneous file)

bb...bb: Size of the source file

S: Resumes processing.

O: Increase the disk space at the target.

KFRB10012-I

The copy of file started, file(*aa...aa*). (L)

The file will be copied.

aa...aa: File name

S: Resumes processing.

KFRB10013-W

The copy of file failed, file(*aa...aa*). (L)

The file copy operation failed.

aa...aa: File name

S: Resumes processing.

KFRB10014-W

The kind of file is invalid, file(*aa...aa*). (L)

The file type is invalid.

aa...aa: File name

S: Resumes processing.

KFRB10015-W

A block of file was broken, file(*aa...aa*). (L)

A file block is damaged.

aa...aa: File name

S: Resumes processing.

KFRB10016-W

A size of file was zero, file(*aa...aa*). (L)

The file size is 0 bytes.

aa...aa: File name

S: Resumes processing.

KFRB10017-W

A copy of file was failed, file(*aa...aa*). (L)

The file copy operation failed.

aa...aa: File name

S: Resumes processing.

KFRB10018-E

The program unable to continue. Please restart this program,
file(*aa...aa*). (L)

The program cannot continue its processing. Restart the program.

aa...aa: File name

S: Terminates processing.

O: Restart HiRDB Datareplicator. We recommend that you recover the erroneous file.

KFRB10019-W

The status of file is hold at last time, file(*aa...aa*). (L)

The previous status of the file header was HOLD during current file checking.

aa...aa: File name

S: Resumes processing.

KFRB10020-W

The last write time of file is different, file(*aa...aa*). (L)

The last update time is different.

aa...aa: File name

S: Resumes processing.

KFRB10021-I

A file problem has occurred. Stopping usage of the file. (L)

A file error occurred. Switchover processing will be performed.

S: Resumes processing.

KFRB10022-I

Stopping usage of file completed. (L)

Switchover processing has been completed.

S: Resumes processing.

KFRB10023-W

Stopping usage of file failed. (L)

Switchover processing failed.

S: Resumes processing.

KFRB10024-E

The logical file name is different, A(*aa...aa*, *bb...bb*), B(*cc...cc*, *dd...dd*). (L)

The logical file name set in the header of the primary file does not match the logical file name set in the header of the secondary file.

aa...aa: Physical file name for the primary file

bb...bb: Logical file name set in the header of the primary file

cc...cc: Physical file name for the secondary file

dd...dd: Logical file name set in the header of the secondary file

S: Cancels startup processing.

O: Check the correspondence between physical and logical file names for the primary and secondary files displayed in the message to determine whether they match the settings for the duplexing definition file. If they do not match, the link target of the symbolic link created as the physical name might be invalid. If the link target is invalid, re-create the symbolic link so that the link target is correct, and then restart Datareplicator.

KFRB11001-E

Invalid command argument.

Usage : *aa...aa* (S)

The command line is invalid.

aa...aa: Command specification method

S: Stops processing.

O: Correct the command line, and then re-execute the command.

KFRB11002-E

Environment variable value invalid, variable name = *aa...aa*. (S)

Specification of the indicated environment variable is invalid.

aa...aa: Environment variable name

S: Stops processing.

O: Correct the environment variable, and then re-execute the command.

KFRB11003-E

Error occurred in *aa...aa* command. (S)

An error occurred while processing the indicated command started by Datareplicator.

aa...aa: Name of the command that was to be started

hdestart: Initialization of the extraction environment failed.

hdeprep: Creation of the extraction definition preprocessing file failed.

pddef: An error occurred in the HiRDB database definition utility.

pdload: An error occurred in the HiRDB database load utility.

S: Stops processing.

O: Eliminate the cause of the error on the basis of the error information that is output by the command that was to be started, and then re-execute the command.

KFRB11004-E

Time out error occurred. (S)

Initialization of the extraction environment failed due to a timeout.

S: Stops processing.

O: Investigate the cause of the timeout error on the basis of the error information that is output by the *hdestart* command, and then re-execute the command.

KFRB11005-E

MSTERRFILE1 exclusion error occurred. (S)

Initialization of the extraction environment failed because a lock error or a timeout occurred during input/output processing on the extraction master error information file (*msterrfile1*).

S: Stops processing.

O: Re-execute the command after evaluating the following:

- Lock error

Check and, if necessary, revise the command executor's permissions and the permissions of the user who executes the extraction master process.

10. Messages

- Timeout

Investigate the cause of the timeout error on the basis of the error information that is output by the `hdstart` command.

10.3 List of system call errors

The table below describes the correspondence of the system call names to the parameter information displayed in the KFRB00864-W and KFRB00865-E messages that are issued when a system call results in an error. When there is no parameter information to be displayed, an asterisk (*) is displayed.

Table 10-2: List of system call errors

Category	System call	Parameter type	Description
File-related errors	close	File name	Closes file
	CloseHandle	File name	Closes file
	CreateFile	File name	Opens file
	DeleteFile	File name	Deletes file
	fclose	File name	Closes a stream
	fcntl	None	Locks file
	fcntl	None	Unlocks file
	fgets	File name	Inputs a character string from the stream
	fopen	File name	Opens a file and concatenates it to the stream
	fstat	None	Obtains file information
	ftruncate	Truncated length	Truncates file to the specified length
	GetFileSize	None	Obtains file information
	ioctl	None	Obtains sector length
	LockFileEx	None	Locks file
	lseek	File name	Moves file pointer
	open	File name	Opens file
	read	File name	Reads data from file
	ReadFile	File name	Reads data from file

Category	System call	Parameter type	Description
	SetEndOfFile	Truncated length	Truncates file to the specified length
	SetFilePointer	File name	Moves the file pointer
	stat	None	Obtains file information
	unlink	File name	Deletes file
	UnLockFileEx	None	Unlocks file
	write	File name	Writes data to file
	WriteFile	File name	Writes data to file
Memory	calloc	Memory allocation size	Allocates memory
	free	None	Frees memory
	GetProcessHeap	None	Allocates memory
	HeapAlloc	Memory allocation size	Allocates memory
	HeapFree	None	Frees memory
	malloc	Memory allocation size	Allocates memory
	realloc	Memory allocation size	Allocates memory
Security	InitializeSecurityDescriptor	None	Initializes the security identifier
	SetSecurityDescriptorDacl	None	Adds security identifier to ACL
File mapping	CreateFileMapping	None	Creates a file mapping object
	MapViewOfFile	None	Executes file mapping
	mmap	None	Executes file mapping
	munmap	None	Releases file mapping
	UnmapViewOfFile	None	Releases file mapping
Time	time	None	Obtains the lapsed time in seconds

Category	System call	Parameter type	Description
Signal	<code>sigaction</code>	None	Changes process action when signal is received
	<code>sigaddset</code>	None	Adds to a signal set
	<code>sigdelset</code>	None	Deletes from a signal set
	<code>sigemptyset</code>	None	Initializes a signal set to null
	<code>sigprocmask</code>	None	Changes the signal list
	<code>SetConsoleCtrlHandler</code>	None	Changes process action when signal is received
Process	<code>atexit</code>	None	Registers a process end-time function

10.4 List of cause codes

Table 10-3 describes the cause codes that are displayed in messages relating to file duplexing. Table 10-4 describes the cause codes that are displayed in the log input error messages. Table 10-5 describes other cause codes.

Table 10-3: List of cause codes in messages relating to file duplexing

Category	Error No.	Description	User's action
Duplexing definition analysis	-1001	There is an error in the syntax.	Check and, if necessary, revise the syntax of duplexing definition, and then initialize Datareplicator.
	-1002	Logical file name is duplicated.	
	-1003	Specified logical file name is invalid.	
	-1004	Physical file name is duplicated.	
	-1005	Specified physical file name is invalid.	
	-1007	Duplexing definition file contains no entry.	
	-1008	Row length exceeded 1,024 bytes.	
	-1009	Operand name is invalid.	
Manipulation of duplexing control file	-1101	Data linkage file contains a syntax error.	To duplex the data linkage file, specify a matching name for both logical and physical files (either primary or secondary file).
	-1102	The duplexing control file is damaged.	Initialize Datareplicator.
	-1104	Attempt was made to access a duplexing control file that is not compatible with the current version.	
	-1105	The <code>file_dupenv</code> operand was specified in the system definition during initialization, but the duplexing control file does not exist.	

Category	Error No.	Description	User's action
	-1106	The <code>file_dupenv</code> operand was omitted in the system definition during initialization, but there is a duplexing control file.	If you are not using the file duplexing facility, delete the duplexing control file, and then initialize Datareplicator.
	-1107	UNIX files cannot be mixed with character special files for physical files.	For the physical files specified in the duplexing definition, use the same file type for both primary and secondary files.
	-1108	The logical file to be added has already been registered in the duplexing control file.	Check and, if necessary, revise the logical file name in the additional queue duplexing definition file that was specified in the <code>hdemodq</code> command so that the logical file name is different from any existing duplex file.
	-1109	The physical file to be added has already been registered in the duplexing control file.	Check and, if necessary, revise the physical file name in the additional queue duplexing definition file that was specified in the <code>hdemodq</code> command so that the physical file name is different from any existing duplex file.
	-1110	Access to the duplexing control file failed.	See the error message that has been output to the syslog file (to the event log in Windows).
Current file copy command	-1301	The sector sizes do not match between source and target in the RAW device environment.	Use the same sector size as with the source.
	-1302	The source file does not belong to HiRDB Datareplicator or is damaged.	Check the source file.
	-1303	The source file has an invalid file size or is damaged.	
	-1304	Target disk space is insufficient.	Check the target disk space.
	-1305	The target disk does not support large files.	Check and, if necessary, revise the maximum file size supported by the target system.

Category	Error No.	Description	User's action
File duplexing facility	-1401	File is damaged.	Execute the current file copy command.
	-1402	Block is damaged.	<p>This error might occur in the situations listed below. If this is the case, you can achieve initial start successfully by specifying the forced initialization option (<code>hdsstart -i -f</code>).</p> <ul style="list-style-type: none"> • An initial start was executed on the target Datareplicator immediately after the target Datareplicator's version was upgraded. • Initialization was performed for the first time after the import information queue file and the import status file were created in character special files. • A status file specified in the <code>statsfile</code> operand in the import environment definition was to be renamed, but a file with the new status file name already exists. <p>If this error occurs during normal operation, execute the current file copy command.</p>
	-1403	File is empty.	Initialization of Datareplicator failed. Eliminate the cause of the error and re-initialize the Datareplicator.
	-1404	During the previous session, the current file copy command did not terminate normally.	Re-execute the current file copy command.
	-1405	Operation cannot continue even in the single-operation mode.	Restart Datareplicator.

Category	Error No.	Description	User's action
	-1406	During the previous session, an error occurred in the primary file.	Execute the current file copy command.
	-1407	During the previous session, an error occurred in the secondary file.	
	-1408	The primary file was placed in HOLD status because its last update time is earlier than that of the secondary file.	
	-1409	The secondary file was placed in HOLD status because its last update time is earlier than that of the primary file.	
	-1410	No current file is available	Execute Datareplicator's data linkage recovery facility.
	-1411	The logical file names do not match between the primary and secondary systems.	Specify the physical file name that constitutes the logical file.
Internal conflict	-2003	Argument is invalid.	Contact the customer support center.
	-2004	Issuance sequence is invalid.	
	-2102	Not found in the FDS management table.	
	-2201	Not enough buffer space.	
	-2202	File descriptor is invalid.	
Other	-2101	Mapped file is damaged.	OS or disk might be unstable. Check for OS or disk errors.
	-2103	Too many files were specified in the Datareplicator definition.	Check and, if necessary, revise the defined value.
Common to all files	-3001	Files are not compatible.	For the source Datareplicator, execute the <code>hdestart -i</code> command; for the target Datareplicator, execute the <code>hdsstart -i -f</code> command.

Table 10-4: List of cause codes that are displayed in the log input error messages

Category	Error No.	Description	User's action
Log input error	-1101	Parameter is invalid.	Contact the customer support center.
	-1105	Protocol is invalid.	Contact the customer support center.
	-1108	Definition is invalid.	Correct the error in the source HiRDB's definition, and then restart the source Datareplicator.
	-1109	Environment variable is invalid.	Correct the error in the environment variable settings, and then restart the source Datareplicator.
	-1113	Memory is insufficient.	Resolve the memory shortage in the environment, and then restart the source Datareplicator.
	-1117	Open error occurred in the log file.	Contact the customer support center.
	-1118	Log start location is invalid.	If the source Datareplicator was initialized while the source HiRDB's linkage was enabled, disable the source HiRDB's linkage, and then initialize Datareplicator. If you are running the facility for recovering the extraction information queue file, this facility cannot be used to restore data because the log file required for recovery has been overwritten. Use another facility, such as the data linkage recovery facility. In all other cases, contact the customer support center.
	-1121	Buffer size is invalid.	Respecify the <code>logiosize</code> operand in the extraction environment definition with a sufficient value, and then restart the source Datareplicator.
	-1143	I/O error occurred.	Contact the customer support center.
	-1144	Log block or data in the log block is invalid.	

Table 10-5: List of other cause codes

Category	Error No.	Description	User's action
File access	-3001	Files are not compatible.	For the source Datareplicator, execute the <code>hdestart -i</code> command; for the target Datareplicator, execute the <code>hdsstart -i -f</code> command.

Appendixes

- A. Detailed Information About HiRDB-Related Datareplicator Support
- B. Datareplicator Reserved Words
- C. Functional Differences Between the UNIX and Windows Editions of Datareplicator
- D. Downgrading Datareplicator
- E. Glossary

A. Detailed Information About HiRDB-Related Datareplicator Support

This appendix lists the OSs that are supported for HiRDB along with the versions of Datareplicator that can be used. It also describes the support provided by Datareplicator for Datareplicator-related HiRDB facilities.

A.1 OSs supported for HiRDB and the versions of Datareplicator that can be used on those OSs

The following table shows the OSs that are supported for HiRDB along with the versions of Datareplicator that can be used.

Table A-1: OSs that are supported for HiRDB along with the versions of Datareplicator that can be used

OS supported for HiRDB	Datareplicator versions supported by the 32-bit edition of HiRDB ^{#1}	Datareplicator versions supported by the 64-bit edition of HiRDB
AIX 5L V5.1	06-00 and later	06-00 and later ^{#2}
AIX 5L V5.2	07-00 and later	07-00 and later ^{#2}
AIX 5L V5.3	07-03 and later	07-03 and later ^{#2}
AIX V6.1	08-03 and later	08-03 and later
HP-UX 11.0	03-00 and later	05-01 and later ^{#2}
HP-UX 11i	06-00 and later	06-00 and later ^{#2}
HP-UX 11i V2 (PA-RISC)	07-04 and later	07-04 and later ^{#2}
HP-UX 11i V2 (IPF)	--	07-01 and later
HP-UX 11i V3 (IPF)	--	08-02 and later
Solaris 8	05-03 and later	05-03 and later ^{#2}
Solaris 9	07-00 and later	07-00 and later ^{#2}
Solaris 10	08-00 and later	08-00 and later ^{#2}
Red Hat Linux 7.1	06-03 and later	--
Red Hat Linux 7.2	06-03 and later	--

OS supported for HiRDB	Datareplicator versions supported by the 32-bit edition of HiRDB^{#1}	Datareplicator versions supported by the 64-bit edition of HiRDB
Red Hat Enterprise Linux AS 2.1	07-00 and later	--
Red Hat Enterprise Linux AS 3 (x86)	07-01 and later	--
Red Hat Enterprise Linux ES 3 (x86)	07-01 and later	--
Red Hat Enterprise Linux AS 4 (x86)	08-00 and later	--
Red Hat Enterprise Linux ES 4 (x86)	08-00 and later	--
Red Hat Enterprise Linux AS 3 (AMD64 & Intel EM64T)	07-03 and later	N
Red Hat Enterprise Linux AS 4 (AMD64 & Intel EM64T)	08-00 and later	N
Red Hat Enterprise Linux ES 4 (AMD64 & Intel EM64T)	08-00 and later	N
Red Hat Enterprise Linux AS 3 (IPF)	--	07-01 and later
Red Hat Enterprise Linux AS 4 (IPF)	--	08-00 and later
Red Hat Enterprise Linux 5.1 Advanced Platform (x86)	08-03 and later	--
Red Hat Enterprise Linux 5.1 (x86)	08-03 and later	--
Red Hat Enterprise Linux 5.1 Advanced Platform (AMD/Intel 64)	08-03 and later	--
Red Hat Enterprise Linux 5.1 (AMD/Intel 64)	08-03 and later	--
Red Hat Enterprise Linux 5.1 Advanced Platform (Intel Itanium)	--	08-03 and later
Red Hat Enterprise Linux 5.1 (Intel Itanium)	--	08-03 and later
Windows 2000	05-02 and later ^{#2}	--
Windows XP Professional	07-01 and later ^{#2}	--
Windows XP x64 Edition	08-00 and later ^{#2}	N
Windows Server 2003	07-00 and later ^{#2}	--

OS supported for HiRDB	Datareplicator versions supported by the 32-bit edition of HiRDB ^{#1}	Datareplicator versions supported by the 64-bit edition of HiRDB
Windows Server 2003 x64 Editions	07-04 and later ^{#2}	N
Windows Server 2003 R2	08-00 and later ^{#2}	--
Windows Server 2003 R2 x64 Editions	08-00 and later ^{#2}	N
Windows Server 2003 (IPF)	--	07-01 and later
Windows Vista Ultimate	08-01 and later	N
Windows Vista Business	08-01 and later	N
Windows Vista Enterprise	08-01 and later	N
Windows Server 2008 Standard	08-03 and later	08-03 and later
Windows Server 2008 Enterprise	08-03 and later	08-03 and later

Legend:

--: The OS is not supported for HiRDB.

N: Datareplicator is not supported.

Note:

This table is applicable for both HiRDB/Single Server and HiRDB/Parallel Server.

#1

The 32-bit edition of HiRDB includes the POSIX library version.

#2

If the HiRDB version is 08-02 or later, the supported Datareplicator versions are 08-01 and later.

A.2 HiRDB facilities and their support by Datareplicator

The table below lists HiRDB facilities that affect Datareplicator operations and the support provided for those facilities by Datareplicator.

The information provided in this table applies to HiRDB 08-03 and Datareplicator 08-02. For the most recent information, see the manuals published online at the following location:

<http://www.hitachi.co.jp/Prod/comp/soft1/manual/common/>

hirdb.htm

Table A-2: HiRDB facilities and the support provided for those facilities by Datareplicator

HiRDB facility			Supporting Datareplicator versions ^{#1}	What happens if a non-supporting Datareplicator version is used
Structure of a table subject to linkage	CREATE TABLE	FIX	Source: 02-00 and later (HiRDB is 03-03 or later)	--
			Target: 01-00 and later	
		SHARE	Source: 02-00 and later	--
			Target: 07-01 and later	Import processing results in an SQL execution error.
		SUPPRESS DECIMAL	Source: 03-00 and later	Values imported to the target database differ from the values in the source database.
			Target: 01-00 and later	--
		WITHOUT ROLLBACK	Source: 08-02 and later	Inconsistencies might occur between the source and target databases. Also, import processing might result in an SQL execution error.
			Target: 08-02 and later	
		REFERENCES	Source: 02-00 and later	--
			Target: 01-00 and later (with conditions ^{#2})	If the conditions are not satisfied, import processing results in an SQL execution error.

A. Detailed Information About HiRDB-Related Datareplicator Support

HiRDB facility		Supporting Datareplicator versions#1	What happens if a non-supporting Datareplicator version is used
	ARRAY	Source: 05-02 and later (HiRDB is 05-05 and later)	Values imported to the target database differ from the values in the source database.
		Target: 05-02 and later	
	NO SPLIT	Source: 05-02 and later	Values imported to the target database differ from the values in the source database.
		Target: 01-00 and later	--
	CHARACTER SET	Source: 08-03 and later	Values imported to the target database differ from the values in the source database.
		Target: 08-03 and later	Unintended import processing is performed on hash-partitioned tables.
	SUPPRESS	Source: 03-00 and later	Values imported to the target database differ from the values in the source database.
		Target: 01-00 and later	--
	COMPRESSED	Source: 08-06 and later	Inconsistencies might occur between the source and target databases. Also, import processing might result in an SQL execution error.
		Target: 08-06 and later	

HiRDB facility				Supporting Datareplicator versions#1	What happens if a non-supporting Datareplicator version is used
		Data type	INTEGER	Source: 02-00 and later	--
			SMALLINT	Target: 01-00 and later	
		DECIMAL	Maximum precision is 29	Source: 02-00 and later Target: 01-00 and later	--
			Maximum precision is 38	Source: 08-03 and later Target: 08-03 and later	
		FLOAT	Source: 02-00 and later Target: 01-00 and later	--	
		SMALLFLT			
		CHARACTER			
		VARCHAR			
		NCHAR			
		NVARCHAR			
		MCHAR			
		MVARCHAR			
		DATE			
TIME	(Not using a leap second)	Source: 02-00 and later Target: 01-00 and later			--

A. Detailed Information About HiRDB-Related Datareplicator Support

HiRDB facility				Supporting Datareplicator versions#1	What happens if a non-supporting Datareplicator version is used
			(Using a leap second)	Source: 02-00 and later Target: 01-00 and later (HiRDB is 08-02 and later)	If the target HiRDB's version is earlier than 08-02, import processing results in an SQL execution error.
		TIMESTAMP		Source: 07-00 and later Target: 07-00 and later	--
		INTERVAL YEAR TO DAY		Source: 02-00 and later Target: 01-00 and later	--
		INTERVAL HOUR TO SECOND			
		BLOB		Source: 02-01 and later Target: 02-01 and later	Environment cannot be configured because the <code>hdeprep</code> command results in an error.
		BINARY		Source: 07-00 and later Target: 07-00 and later	--
	Abstract data type	SGMLTEXT		Source: 05-00 and later Target: 05-00 and later	--
		FREEWORD		Source: 08-02 and later Target: 08-02 and later	--

HiRDB facility				Supporting Datareplicator versions#1	What happens if a non-supporting Datareplicator version is used	
				XML	Source: 08-02 and later Target: 08-02 and later	--
				Abstract data type defined by the user by using CREATE TYPE	Source: Not supported Target: Not supported	Environment cannot be configured because the <code>hdeprep</code> command results in an error.
	Index with at least one column whose definition length is 256 bytes or greater				Source: Not supported	Source system's environment cannot be configured because the <code>hdeprep</code> command results in an error.
					Target: 01-00 and later	--
	Trigger				Source: 07-00 and later Target: 07-00 and later	--
Manipulation of a table subject to linkage	PURGE TABLE	Non-partitioned table or row-partitioned table in a server			Source: 04-00 and later (HiRDB is 04-04 and later)	If the source HiRDB's version is earlier than 04-04, Purge table is not linked to the target database.
					Target: 04-00 and later	
			Table that is row-partitioned among multiple servers		Source: Not supported	Purge table is not linked to the target database.
					Target: 04-00 and later	
	INSERT				Source: 02-00 and later Target: 01-00 and later	--

A. Detailed Information About HiRDB-Related Datareplicator Support

HiRDB facility		Supporting Datareplicator versions#1	What happens if a non-supporting Datareplicator version is used
UPDATE	Performing a concatenation operation on BLOB type or BINARY type with a length of 32,001 bytes or greater	Source: 06-01 and later Target: 06-01 and later	Values imported to the target database differ from the values in the source database.
	Performing backward deletion updating on BLOB type or BINARY type with a length of 32,001 bytes or greater	Source: 08-01 and later Target: 08-01 and later	Values imported to the target database differ from the values in the source database.
	Other	Source: 02-00 and later Target: 01-00 and later	--
DELETE		Source: 02-00 and later Target: 01-00 and later	--
Overall SQL statements	USER (using USER as the update value for the source database) (using USER as the update value for the target database)	Source: 02-00 and later	--
		Target: Not supported	The same value as the update value for the source database is imported.
	CURRENT_DATE value function CURRENT_TIME value function (using the CURRENT_DATE or CURRENT_TIME value function as the update value for the source database) (using the CURRENT_DATE or CURRENT_TIME value function as the update value for the target database)	Source: 02-00 and later Target: 01-00 and later	--
	CURRENT_TIMESTAMP value function (using the CURRENT_TIMESTAMP value function as the update value for the source database) (using the CURRENT_TIMESTAMP value function as the update value for the target database)	Source: 02-00 and later Target: 07-00 and later	--

HiRDB facility		Supporting Datareplicator versions ^{#1}	What happens if a non-supporting Datareplicator version is used		
HiRDB facilities that affect data linkage	Space conversion facility		Source: 02-00 and later	--	
			Target: 05-03 and later	Unintended import processing is performed on hash-partitioned tables.	
	Character code classification	sjis		Source: 02-00 and later Target: 01-00 and later	--
		euc		Source: 02-00 and later	--
				Target: 01-02 and later	Import environment cannot be configured.
		korea		Source: Not supported Target: Not supported	Environment cannot be configured.
		lang-C		Source: 02-00 and later Target: 01-00 and later	--
		utf-8	Within the range of UCS-2	Source: 07-01 and later Target: 07-01 and later	Environment cannot be configured.
			Within the range of UCS-4	Source: 07-01 and later	Extraction environment cannot be configured.

A. Detailed Information About HiRDB-Related Datareplicator Support

HiRDB facility				Supporting Datareplicator versions ^{#1}	What happens if a non-supporting Datareplicator version is used	
				Source: 07-01 and later (not supported if the character code classification differs between the source and target databases)	A character whose length is 4 bytes or greater is converted to an undefined character, and then imported.	
				chinese	Source: Not supported Target: Not supported	Environment cannot be configured.
	System switchover	When HiRDB and Datareplicator are running on the same server	Monitor mode, server mode, or user server hot standby	IP addresses inherited	Source: 02-00 and later Target: 01-00 and later	--
				IP addresses not inherited	Source: 06-02 and later	Source system does not function when system switchover occurs.
					Target: 01-00 and later	--
					Source: 06-02 and later	Source system does not function when system switchover occurs.
				Target: 01-00 and later	--	
			Rapid system switchover facility or standby-less system switchover (1:1) facility	Source: 06-02 and later	Source system does not function when system switchover occurs.	
				Target: 01-00 and later	--	
			Standby-less system switchover (effects distributed) facility	Source: 07-01 and later (Windows edition is not supported)	Source system does not function when system switchover occurs.	
Target: 01-00 and later	--					

HiRDB facility			Supporting Datareplicator versions ^{#1}	What happens if a non-supporting Datareplicator version is used
	When HiRDB and Datareplicator are running on separate servers		Source: Not supported	Extraction environment cannot be configured.
			Target: 05-01 and later	Once system switchover has occurred, connection with HiRDB is lost and import processing stops.
Real Time SAN Replication	All synchronous method, all asynchronous method, or log-only synchronous method	Main site	Source: 02-00 and later Target: 01-00 and later	--
		Remote site	Source: Not supported	No update information is extracted.
			Target: 01-00 and later	--
DNS support			Source: 02-00 and later Target: 01-00 and later	--
Linkage facility	When using Datareplicator's import transaction synchronization facility		Source: 07-04 and later Target: 07-04 and later	There are no effects.
	When using Datareplicator's data linkage recovery facility		See Table 9-10 <i>Combinations of the versions and products that support data linkage recovery via unload log files.</i>	The data linkage recovery facility is not supported.

A. Detailed Information About HiRDB-Related Datareplicator Support

HiRDB facility			Supporting Datareplicator versions#1	What happens if a non-supporting Datareplicator version is used
	Inner replica facility	Linking updates to current RDAREAs only	Source: 06-01 and later	Updates to original RDAREAs and all replica RDAREAs are subject to linkage processing.
			Target: 01-00 and later	--
		Linking updates to original RDAREAs and all replica RDAREAs	Source: 02-00 and later Target: 01-00 and later	--
Utilities	pdload -d		Source: Not supported Target: Not supported	Import processing results in an SQL execution error. Another possibility is that existing rows are intermixed with rows inserted by pdload.
	pdload -lp pdload -ln	Source: Not supported	Update information is not extracted.	
		Target: 01-00 and later	--	
	pdrbal	Source: Not supported	Data is not extracted from any added back-end server.	
		Target: 01-00 and later	--	
pdroreg -la	Source: Not supported	Import processing results in an SQL execution error. Another possibility is that existing rows are imported twice.		

HiRDB facility		Supporting Datareplicator versions ^{#1}	What happens if a non-supporting Datareplicator version is used
		Target: 01-00 and later	--
UAP execution environment	No-log mode	Source: Not supported	No update information is extracted.
		Target: 01-00 and later	--
	PDCWAITTIME PDSWAITTIME	Source: --	There are no effects.
		Target: Not supported	If 0 is specified in PDCWAITTIME or PDSWAITTIME, SQL statements are executed on the target database.

Legend:

--: There are no unsupported versions.

#1

If a supported version is indicated, extraction and import processing are supported by that version and all versions subsequent to it. If there are conditions, the conditions take precedence over the version information.

#2

The transaction-based import method must be used, and updates to the source database must be performed on referencing tables and referenced tables in this order.

B. Datareplicator Reserved Words

The table below lists the reserved words used in Datareplicator definitions. Except as noted below, these reserved words are not case-sensitive. Note that all operand names used in the Datareplicator extraction, import, and update information definitions are also reserved words.

If any of these character strings is used in extraction Datareplicator definitions, target Datareplicator definitions, or update information definitions, it must be enclosed in double quotation marks ("").

Table B-1: Datareplicator reserved words

Reserved word		Used in source Datareplicator definitions	Used in target Datareplicator definitions	Used in update information definitions
A	adt	Y	Y	Y
	and	Y	--	--
	attr	--	--	Y
B	binary	--	--	Y
	by	Y	Y	--
C	char	--	--	Y
	check	Y	Y	--
	comp	--	--	Y
	const	--	Y	--
	construct	Y	Y	Y
D	divide	--	Y	--
E	extract	Y	--	Y
	extract_date	--	Y	--
	extract_time	--	Y	--
F	field	--	--	Y
	filetype	--	--	Y
	flike	Y	--	--
	float	--	--	Y

	Reserved word	Used in source Datareplicator definitions	Used in target Datareplicator definitions	Used in update information definitions
	format	--	Y	--
	from	Y	Y	--
G	group	--	Y	--
H	hash	--	Y	--
	having	--	Y	--
	Hdsdefserv ^{#1}	--	Y	--
	hdsmain ^{#1}	--	Y	--
	hdsreflect ^{#1}	--	Y	--
	hdssqle ^{#1}	--	Y	--
	hdstcpmst ^{#1}	--	Y	--
I	in	Y	Y	--
	ins	--	--	Y
	into	--	Y	--
	is	--	Y	--
	ivl	--	--	Y
K	key	Y	--	Y
L	lib	Y	Y	Y
	load	--	Y	--
N	nchar	--	--	Y
	name	--	Y	--
	nocodecnv	--	Y	--
	none	Y	Y	--
	not	--	Y	Y
	not_null_unique	Y	Y	--
	null	--	Y	Y

B. Datareplicator Reserved Words

Reserved word		Used in source Datareplicator definitions	Used in target Datareplicator definitions	Used in update information definitions
O	or	Y	--	--
	other	--	Y	--
	othergrp	--	Y	--
P	pack	--	--	Y
	packns	--	--	Y
	pdm	--	--	Y
	position	--	--	Y
Q	que	--	--	Y
R	reflect_date	--	Y	--
	reflect_kind	--	Y	--
	reflect_time	--	Y	--
	reptype	Y	Y	Y
	restruct	--	--	Y
S	send	Y	--	--
	seq_no	--	Y	--
	sqlconvopt1	--	Y	--
	sqlconvopt2	--	Y	--
T	through	Y	Y	Y
	timestamp	--	Y	--
	to	Y	Y	Y
	trngroup	--	Y	--
U	ukey	Y	--	Y
	unique	Y	Y	--
	unpack	--	--	Y
	unpackns	--	--	Y
	uocname	--	--	Y

Reserved word		Used in source Datareplicator definitions	Used in target Datareplicator definitions	Used in update information definitions
	uocxxx ^{#2}	--	Y	--
	upd	--	--	Y
	update	--	--	Y
W	where	Y	--	--

Y: Reserved word.

--: Not a reserved word.

Note:

The reserved words are not case-sensitive, except as noted in footnote 1 below. In the case of the reserved word `by`, `by`, `bY`, `By`, and `BY` are all reserved words.

#1: This is a reserved word only when it is in all lowercase letters.

#2: `xxx` represents an unsigned three-digit integer.

C. Functional Differences Between the UNIX and Windows Editions of Datareplicator

The following table describes the functions that differ between the UNIX and Windows Datareplicators.

Table C-1: Functions that differ between the UNIX and Windows Datareplicators

Item	UNIX Datareplicator	Windows Datareplicator
File system	<ul style="list-style-type: none"> UNIX file system Regular files and character special files are supported.	<ul style="list-style-type: none"> FAT, NTFS There is no file system that corresponds to character special files.
Communications protocols	<ul style="list-style-type: none"> TCP/IP OSI (for establishing data linkage with a mainframe database) 	<ul style="list-style-type: none"> TCP/IP
Permitted number of target Datareplicators per source Datareplicator	The permitted range is determined by the <code>sendcontrol</code> operand value in the extraction system definition. When <code>nodemst</code> is specified: 1-64 When <code>sendmst</code> is specified: 1-4096	The permitted range is determined by the <code>sendcontrol</code> operand value in the extraction system definition. When <code>nodemst</code> is specified: 1-63 When <code>sendmst</code> is specified: 1-4096
Number of target identifiers (specification for source Datareplicator)	Same as number of <code>sendidxx</code> operands in extraction system definition (<code>xx: 01-64</code>) Or, same as number of <code>sendidxxx</code> operands in extraction system definition (<code>xxx: 0001-4096</code>)	Same as number of <code>sendidxx</code> operands in extraction system definition (<code>xx: 01-63</code>) Or, same as number of <code>sendidxxx</code> operands in extraction system definition (<code>xxx: 0001-4096</code>)
Maximum number of transmission processes that can be active (specification for source Datareplicator)	1-1024	1-63
Range of data linkage identifiers (specification for target Datareplicator)	Number of <code>dsidxxx</code> operands in import system definition (<code>xxx: 001-128</code>)	Number of <code>dsidxxx</code> operands in import system definition (<code>xxx: 001-063</code>)
UOC creation method	Executable file	DLL file

Item	UNIX Datareplicator	Windows Datareplicator
Message output destinations	<ul style="list-style-type: none">• Standard error output• Error information files• Activity trace files• syslog file	<ul style="list-style-type: none">• Standard error output• Error information files• Activity trace files• Event log
Control for output of information in the extraction node master error information to the syslog file or event log	Controlled by the <code>node_syslogout</code> operand	Controlled by the <code>syslogout</code> operand

D. Downgrading Datareplicator

This appendix describes how to downgrade Datareplicator.

D.1 Differences in downgrading procedures between product models

Datareplicator's product model depends on the version (such as Version 8 or Version 7). To use Datareplicator with a HiRDB/Parallel Server, make sure that the system manager and all back-end servers use the same version of Datareplicator.

D.2 Downgrading procedure

This subsection describes how to downgrade Datareplicator.

1. Make sure that all extraction and import processing has been completed at the source and target Datareplicators (there is no more unextracted or unimported data).
2. Terminate HiRDB and Datareplicator normally. If you are using a HiRDB/Parallel Server, terminate all servers normally.
3. Uninstall Datareplicator.
4. Install the Datareplicator to which you are downgrading.
5. If you changed the following Datareplicator items at the time of upgrading, reset them to their previous status before you upgraded:
 - Various definition files
 - Gaiji mapping table
 - UOC routines
6. Initialize Datareplicator.
7. Start HiRDB and Datareplicator.

If this method cannot downgrade the Datareplicator, initialize the Datareplicator with the procedure described in *6.5.6(1) Handling procedure for re-creating the target database*, and then downgrade it.

D.3 Notes after downgrading

Note the following about Datareplicator after downgrading:

- If product models are different before and after downgrading, or if the file compatibility described in *2.11.3 Notes on upgrading* is missing, make sure that an initial start is executed on Datareplicator immediately after downgrading.

- Even if you have downgraded UNIX Datareplicator to a 64-bit version of Datareplicator, make sure that UOC routines are preprocessed, compiled, and linked in a 32-bit environment. Executable files that use 64-bit libraries cannot be used. If use of such a UOC routine is attempted, the operating results cannot be guaranteed.

E. Glossary

abstract data type

A data type for large and complicated data, such as multimedia data or three-dimensional data. Abstract data types comply with the SQL3 standard with object-oriented extensions. These data types enable users to define unique data with a complicated structure as well as procedures for its manipulation.

Datareplicator supports HiRDB's `SGMLTEXT`, `FREEWORD`, and `XML` abstract data types for data linkage. To use the `SGMLTEXT` and `FREEWORD` types, you must have HiRDB Text Search Plug-in or HiRDB XML Extension. HiRDB XML Extension is required to use the `XML` type,.

ADT (Abstract Data Type)

See *abstract data type*.

advanced queue

A message queuing function provided by Oracle. For details, see the Oracle documentation.

application

Collective name for HiRDB application processing. A program created as an application is referred to as the application program (or user application program (UAP)).

association

Logical communication route used to establish communication using the OSI protocol.

back-end server

A server component of a HiRDB/Parallel Server. A back-end server provides functions for accessing and locking databases, and functions for receiving and executing sort/merge tasks from other back-end servers.

BES (Back-end Server)

See *back-end server*.

BLOB (Binary Large Object)

Acronym for Binary Large Object, which refers to large data, such as documents, images, and audio data.

client/server

Term indicating the relationship of program-to-program communications. A client issues an application processing request, and a server accepts the request and executes

the application. Client/server indicates a relative relationship between such programs. The term client/server can be applied to programs as well as to hardware (such as workstations and personal computers).

cluster key

A table column that is specified as the key for storing rows in ascending or descending order of the values in the specified column.

data definition language

A language for defining the organization and contents of a database.

data dictionary

A dictionary that contains a database's design specifications, such as the database table structures, column definitions, and index definitions. A single database is divided for storage purposes among multiple servers (in the case of HiRDB/Parallel Server) and managed collectively.

data dictionary LOB RDAREA

An RDAREA for storing stored procedures. The two types of data dictionary LOB RDAREAs are a type for storing the definition sources of stored procedures and a type for storing their objects.

data manipulation language

A language for specifying how a database will be manipulated by an application program.

data warehouse

A mechanism for managing a key system's data in such a manner that the data can be used easily by end users in a data system.

A key application's databases are designed to achieve efficient, high-speed data update processing. For the data application's databases, on the other hand, managing data in an easy-to-handle format according to the end user's purposes is more important than processing efficiency and speed. A data warehouse provides data applications with an effective data utilization mechanism. When a data warehouse is constructed, the data accumulated by key applications can be used efficiently by data applications, such as for end users' decision making support and application analysis. Additionally, if a database is created to accumulate updated data in chronological order (a time-ordered database), trends and progressive changes in information can also be analyzed.

Datareplicator and HiRDB Dataextractor's replication facility enable a HiRDB server to be used as a database in a data warehouse.

deadlock

Deadlock occurs when multiple transactions are competing for the same resources and

are waiting for each other to release their resources.

dictionary server

A server component of a HiRDB/Parallel Server. A dictionary server is used to manage data dictionaries.

DS (Dictionary Server)

See *dictionary server*.

embedded UAP

A UAP in which SQL statements are written directly into a source program that is written in a high-level language (C or COBOL).

FES (Front-end Server)

See *front-end server*.

FIX attribute table

A table whose rows are fixed in length.

flexible hash partitioning

Method for partitioning a table that uses a hash function to effect uniform distribution of the table's column values among RDAREAs. The column used as the basis for partitioning a table's rows is called the partitioning key.

FREWORD type

See *abstract data type*.

front-end server

A server component of a HiRDB/Parallel Server. A front-end server provides functions for establishing connections with clients, distributing processing to back-end servers, and collecting the processing results.

global buffer

Buffer used for input/output of data between a HiRDB and the HiRDB files.

global buffer pool

A set of global buffer sectors of the same size. A global buffer is used for input/output of data stored in RDAREAs on the disk. A global buffer pool is allocated in shared memory. Global buffers are always allocated for RDAREAs and indexes.

host

A single computer (machine) connect to a network on which HiRDB is run. In the case of a HiRDB/Parallel Server, servers are distributed to multiple hosts. In the case of a multi-HiRDB, one host consists of multiple HiRDBs.

index

An index created on the basis of specific columns in a table in order to improve data retrieval efficiency. An index is a structure assigned to one or more columns to provide a key for searching the table; an index consists of the key and key values. A key is a column name that indicates the contents of the column, while key values are the data values set in the column.

The two types of indexes are single-column indexes and multicolumn indexes. A single-column index is based on indexing only one column of a table. A multicolumn index is based on indexing multiple columns of a table as a single index.

interface area

An area used to pass information between HiRDB and a UAP.

JP1

Collective name for a group of products equipped with various facilities such as batch job operation, automatic system operation, forms output control, and file backup. JP1 is used to implement automatic system operation and energy conservation. The JP1 products include JP1/Automatic Operation Monitor, which automates system operation, and JP1/System Event Service, which manages events issued by programs.

key range partitioning

A method of partitioning a table. Key range partitioning divides the table into groups of rows on the basis of ranges of values in a specified column of the table. The column that is the basis for row partitioning is called the partitioning key.

linkage recovery backup file

File created for each HiRDB file by using the backup command (`pdfbackup`) to back up HiRDB files containing unload log files from the HiRDB file system area. Linkage recovery backup files are used by the data linkage recovery facility.

LOB data

Large, variable-length data, such as documents, images, and audio LOB data is stored in user LOB RDAREAs. Data loading and database reorganization can be executed for LOB RDAREAs separately from the user RDAREAs containing the LOB column structure base table.

MGR (System Manager)

See *system manager*.

multi-HiRDB

A mode in which multiple HiRDBs run on a single host. Each HiRDB is operated separately. A separate HiRDB administrator must be registered for each HiRDB.

node

A communication control unit and the system connected to it that constitute a network component in a distributed system. A node refers to a host or a HiRDB system.

null value

A special value indicating that no value has been set.

partitioning key index

An index whose primary component column is the one for which storage conditions are specified for a partitioned table. It includes the following indexes:

- A single-column index created on the basis of the partitioning key
- A multicolumn index created on the basis of multiple columns beginning with the partitioning key

An index that is not a partitioning key index is called a non-partitioning key index.

plug-in

A package product that extends HiRDB's data management capabilities. HiRDB's abstract data types allow a wide variety of data to be retrieved. HiRDB provides the following plug-ins:

- HiRDB Text Search Plug-in
- HiRDB Image Search Plug-in
- HiRDB Spatial Search Plug-in

port number

A number assigned to a service that is available in a network.

process

The unit of program execution in a work area that is created when application software, such as a UAP or HiRDB, is executed under the operating system. Virtual space and time-shared CPU resources are allocated to each process.

RD-node

A DBMS identified by a location in a network that is connected with the distributed database facility. A node name used to specify a DBMS at the target server in the distributed network is referred to as the RD-node name. The RD-node name is equivalent to the resource name under the OSI-RDA protocol.

regular file

A simple UNIX dynamic file. Because it can be created and deleted with UNIX commands, a regular file is suitable for creating a test database.

repetition column

A column composed of multiple elements in a relational database table. Because a repetition column can store multiple elements in a single cell, what were conventionally separate tables can be combined into a single table.

Datareplicator data linkage handles a repetition column as a single column. Data linkage is not supported for each element of a repetition column.

replication facility

A facility that enables data to be imported from one database into a database in another system by linking databases between a mainframe and HiRDB, or between two separate HiRDBs. HiRDB's replication facilities include the data linkage facility (HiRDB Datareplicator) and the data extraction and import service facility (HiRDB Dataextractor).

row partitioning

Division of a table, index, or LOB column into multiple partitions for storage in multiple user RDAREAs or user LOB RDAREAs. When a table is row partitioned, its indexes can also be partitioned on the same basis as the table partitions. If a table contains a LOB column, the LOB column can be divided and stored in multiple user LOB RDAREAs on the same basis as the table partitions.

server

In the HiRDB manuals, a server refers to a component unit constituting a HiRDB system.

SGMLTEXT type

See *abstract data type*.

statistics log file

A file for storing statistical information (statistics log) output by HiRDB.

status file

A file used to store system status information that is needed in order to restart HiRDB.

superuser

The UNIX user with the highest privileges. A superuser can access all files in the UNIX file system. The superuser name is `root`.

synchronization point

A punctuation point of transaction processing. A commit refers to the synchronization point processing that indicates that transaction processing terminated normally, while a rollback refers to the synchronization point processing that is executed in order to ignore transaction processing because it did not execute successfully.

synchronization point dump file

A file containing HiRDB management information needed to restart HiRDB or restore a database. The system log is separated by synchronization points at specified intervals, and HiRDB management information is saved at each synchronization point. In the event of a failure, the system is restored by using the most recent synchronization point dump and the system log information that has been collected since that synchronization point dump occurred. This method reduces system recovery time because there is no need to read the collected system log from the beginning.

system files

A collective name for the following types of files:

- System log files
- Synchronization point dump files
- Status files

system log file

A file used to store historical information about database update processing.

system manager

A server component of a HiRDB/Parallel Server. The system manager's principal role is to control the execution of commands and utilities.

system RDAREA

A collective name for the following types of RDAREAs:

- Master directory RDAREA
- Data directory RDAREA
- Data dictionary RDAREAs

TCP/IP

A protocol developed by ARPANET, a project of Defense Advanced Research Projects Agency (DARPA). The TCP/IP protocol is used mainly with LANs.

transaction

To maintain data conformity, processing such as reading data from a file, and then writing the modified data (update processing) is handled as a unit and is not divided into segments. This unit of processing is called a transaction. The result of transaction processing is always either valid or invalid.

trigger

An operation that causes a predefined SQL statement to be executed automatically when an operation (updating, insertion, or deletion) is performed on a table. Triggers

can be used to update tables automatically when a related table is updated.

UAP

An application created as a program; also referred to as an application program.

unload log file

A file created by unloading a system log file.

unload statistics log file

A file created by unloading the contents of a statistics log file.

update information

Information indicating the details of updating operations performed on the source database.

user LOB RDAREA

An RDAREA for storing BLOB data (large data, such as documents, images, and sounds).

view table

A virtual table defined by selecting rows and columns from an actual table (which is called the base table).

volume

The unit of allocation for storing segments.

work table file

A file used to temporarily store information that is required in order to execute SQL statements.

XML type

See *abstract data type*.

Index

A

abbreviations defined vi
abstract data type 24, 1370
activity trace collection process 112, 129
activity trace file 109, 126, 366, 369
 editing 870
 handling 619, 669
activity trace information collection interval 439, 502
activity trace, collecting 114, 136
ADM 9, 37, 296
ADT 1370
advanced queue 1370
agent process 154
allocation file type 456, 526
application 1370
 of bidirectional updating system to
 hierarchical system 304
 of unidirectional updating system to
 hierarchical system 303
 to hierarchical system, designing data linkage
 system for 303
association 1370
attr redefined field attribute(length) 562
audit trail, notes when not collecting import
processing-related 938
authorization identifier 477, 546, 552
automatic system switchover 697

B

back-end server 1370
BES 1370
besdef (server name) 422
bidirectional updating system 302
 application to hierarchical system 304
BLOB 1370
breakmode 516
breaktime 516
buffer

 designing 322

 used for transmission, designing 325

by 'uoc name' 551

by column data editing UOC function identifier 543

C

cause codes, list of 1340
channel connection 298
character code conversion 263
 by target Datareplicator 264
 from EBCDIK/KEIS to EUC, rules for 266
 from EBCDIK/KEIS to JIS8/Shift JIS, rules
 for 265
 from EUC to JIS8/Shift JIS, rules for 272
 from JIS8/Shift JIS or EUC to EBCDIK/KEIS,
 rules for 270
character code set
 between source and target databases,
 correspondence of 264
 designing for 262
 types of 262
character encoding conversion 95
 from EBCDIK to JIS8, rules for 269
 from JIS8 to EBCDIK, rules for 269
character set, level 3 and 4 93
check 478
check_pending (import environment definition) 531
client/server 1370
cluster key 1371
Cluster Server 698
cm_errno_check (extraction system definition) 447
Cm2/Extensible SNMP Agent 45
cmtintvl 518
cmwaittime 434
column data editing UOC function identifier 543
column data editing UOC routine 137, 944
 creation procedure (UNIX) 946
 creation procedure (Windows) 949
 list of header files for 961

- notes on creating 962
- samples of 963
- syntax for function used with 952
- column name 477
- command log file 317, 366, 369, 391
 - file settings used at source Datareplicator 322
 - handling 620, 671
 - used in extraction processing 110
 - used in import processing 127
- command start method 383, 660
- commands, list of 745
- COMMIT issuance interval 503, 529
 - for import processing, designing 377
- commit_wait_time 503, 529
- commitment_method 499
- common definition section 422
- commondef 421, 422
- communication wait time 434
- communications environment (UNIX)
 - /etc/hosts 66
 - /etc/inetd.conf 66
 - /etc/services 66, 69
 - /etc/services for target Datareplicator, example of 70
 - for target Datareplicator, example of 69
 - setting up 66
 - XNF network definition 69
- communications environment (Windows), setting up 89
- communications protocol 298
- configuration, changing 677
 - when import transaction synchronization facility is used 689
- connection retries count 329, 468
- connection retry interval (transmission environment definition) 468
- connection-based termination event 208
- connection_accept_hostname (extraction system definition) 443
- connection_accept_service (extraction system definition) 444
- connection_accept_waittime (extraction system definition) 444

- connection_retry_time (extraction system definition) 444
- const initial-value 541
- control_reference_trigger (import environment definition) 531
- control_trigger 530
- conventions
 - abbreviations vi
 - diagrams xii
 - fonts and symbols xiii
 - KB, MB, GB, and TB xvi
 - permitted characters xv
 - version numbers xvii
- conversion rule for space character 265, 266
- current file, copying 755, 820

D

- data definition language 1371
- data dictionary 1371
- data dictionary LOB RDAREA 1371
- data linkage 2, 9
 - application system for 4
 - by selecting rows to be sent 103, 240
 - by using UOC routine 104, 242
 - databases supported for 16
 - definition for inner replica 119
 - designing pattern 224
 - for table with trigger set 134
 - from HiRDB to mainframe database 11
 - from HiRDB to mainframe database, designing 292
 - from mainframe database to HiRDB using SAM file 14
 - from mainframe database to HiRDB using SAM file, designing 299
 - from mainframe database to HiRDB using SAM file, operation procedure of 593
 - from mainframe database to HiRDB, designing 296
 - from n tables to one table 102, 238
 - from one HiRDB to another 9
 - from one HiRDB to another HiRDB, designing 285
 - from one table to n tables 102, 234

- linking update data for concatenation operation 117
- mechanism of 9
- pattern 100
- RDAREA using inner replica facility 118
- relationship to database type 17
- systems whose data can be linked 4
- tables supported for 21
- to table with different format 100, 229
- to table with same format 100, 225
- data linkage facility 2, 99
- data linkage file 108, 318, 322, 340, 624
 - handling 619
- data linkage identifier 289, 452, 493, 837, 841, 854
- data linkage recovery
 - via system log file 1012
 - via unload log file 1023
- data linkage recovery facility
 - commands provided by 1048
 - procedure after execution of 1053
- data linkage recovery facility 1 (transaction retrieval phase) 1036
- data linkage recovery facility 2 (extraction queue creation phase) 1036
- data linkage system 2, 4
 - advantages of 3
 - combinations for 9
 - conducting bidirectional updating among more than two systems 310
 - construction procedure 42
 - operating procedure 590
 - terminology for 4
 - that links multiple systems 305
- data linkage system mode
 - designing 302
 - types of 302
- data manipulation language 1371
- data type
 - for tables supported for data linkage 23
 - supported by Datareplicator 23
 - when source database is HiRDB, correspondence of 255
 - when source database is XDM/SD E2, correspondence of 256
 - when source database is XDM/SRDE2, correspondence of 258
 - when the source database is ADM, correspondence of 259
 - when the source database is PDM2 E2, correspondence of 260
 - when the source database is RDB1 E2, correspondence of 261
 - when the source database is TMS-4V/SP, correspondence of 261
- data warehouse 1371
- database
 - relational 16
 - relational-type 16
 - structured 16
 - structured-type 16
- database definition utility 248
- database extraction/import service facility 2
- database load utility 248
- database recovery utility 248
- database reorganization utility 248
- database structure modification utility 248
- database terminology, correspondence of 19
 - column 20
 - component 20
 - database 19
 - dataset 19
 - DBM 19
 - field 20
 - record 20
 - record occurrence 20
 - record type 19
 - row 20
 - schema 19
 - segment 19
 - segment occurrence 20
 - table 19
- database type 16
 - relationship to data linkage 17
- Datareplicator
 - can be used on OSs supported for HiRDB, version of 1348
 - downgrading 1368
 - products associated with 44

- purpose of 2
 - reserved words of 1362
 - support provided for HiRDB facilities by 1350
 - uninstalling (UNIX) 49
- Datareplicator (UNIX) installation 48
 - execution of 49
 - preparations before 48
 - server machine subject to 48
- Datareplicator (Windows) installation 71
 - execution of 72
 - preparations before 71
 - server machine subject to 71
 - services during 73
- Datareplicator agent 148
 - changing settings of 162
 - displaying status of 162, 810
 - manipulating settings of 807
 - operating 161
 - setting up (UNIX) 156
 - setting up (Windows) 156
 - starting 162, 809
 - terminating 162, 812
- Datareplicator definition
 - examples of 567
 - for individual server machines 420
 - organization of 414
 - overview of 414
- Datareplicator file system area 172
 - displaying status of 826
 - initializing 821
- Datareplicator file system area name 456, 526
- Datareplicator support, HiRDB-related 1348
- dataset name 562, 564
- DB connect retry count 525
- DB connect retry interval 525
- db_connect_retry_interval 525
- db_connect_retry_number 525
- dblocale 432, 496
- DBM name 562, 564
- DBM-name.dataset-name 562
- deadlock 1371
- defined information
 - modifying (extraction definition) 476
 - modifying (extraction environment definition) 450
 - modifying (extraction system definition) 425
 - modifying (import definition) 540
 - modifying (import environment definition) 510
 - modifying (import system definition) 490
 - modifying (transmission environment definition) 461
- definition rules 419
 - for file-specific format 420
 - for set format 419
 - for syntax format 420
- defmerge 518
- defshmsize 521
- delay monitoring facility 180
- delay period 180
- delay start 383
- devicexx 456, 526
- diagram conventions xii
- dictionary server 1372
- directory for monitoring
 - setting up 155
 - specifying 829
- directory structure (UNIX) 50
 - of source Datareplicator 51
 - of target Datareplicator 54
- directory structure (Windows) 76
 - of source Datareplicator 77
 - of target Datareplicator 80
- discintvl 498, 529
- disconnect issuance interval 498, 529
- DISCONNECT issuance interval for import processing, designing 376
- disk resource, designing 339, 390
- DLL file 908, 949, 970
- DS 1372
- dsid 452
- dsidxxx 493
- duplexed file, displaying status of 756, 824
- duplexing control file 318, 322, 366, 370
- duplexing definition (definition syntax) 487, 566
- duplexing definition file 317, 340
 - file settings used at source Datareplicator 321

file settings used at target Datareplicator 369
 target Datareplicator's disk resources 390
 used by target Datareplicator 366

E

EBCDIC/KEIS 262
 ebcdic_type 521
 EBCDIK 263
 EBCDIK/KEIS 262
 editbufsize 470
 embedded UAP 1372
 entry 66
 environment setup 43
 environment variables (UNIX)
 for source Datareplicator 58
 for target Datareplicator 61
 specifying 58
 environment variables (Windows)
 for source Datareplicator 84
 for target Datareplicator 85
 specifying 84
 errfile_unique 431
 errfilesz 432, 496
 error
 actions after correcting 992, 1001
 skipping using import suppression 138
 error handling method
 source Datareplicator 988
 target Datareplicator 998
 error handling procedure 985
 at target system 993
 source Datareplicator 986
 target Datareplicator 996
 user own coding routine 995, 1003
 error information file 108, 126
 handling 616, 667
 error monitoring interval 433
 EUC 263
 EUC code system 433, 497
 EUC Kanji encoding 263
 event
 issuing 207
 to reset data-transmission count 208
 to reset import processing count 208
 event code 208, 753
 defining 208
 specification range of 335
 event control table
 conditions of 336
 designing 335
 examples of 337
 structure of 336
 event facility 206
 event number 471
 event start method 384, 660
 event termination 656
 eventcntreset
 import environment definition 518
 transmission environment definition 473
 eventretbl 517
 eventretrn 517
 eventspd 518
 eventsync (transmission environment definition) 473
 eventtbl 517
 eventtrn 517
 except_suppress 436, 498
 ext_wait_interval 454
 extdef 428
 extinfunum 434
 extract authorization-identifier.table-identifier 476
 extract DBM-name.dataset-name 564
 extract_delay_limit_time (extraction environment definition) 455
 extract_init 524
 extract_level 454
 extract_tselector 465
 extracted column
 of FLOAT type 253
 of SMALLFLT type 253
 extracting status 625
 extraction 107, 113
 designing procedure 322
 service 73
 extraction command process 112
 extraction completed status 625
 extraction definition 16, 475
 extraction definition file 109, 317, 321, 340
 converting again 611

- handling 611
 - extraction definition filename 762
 - extraction definition preprocessing file 109, 317, 321, 340
 - checking validity of 109
 - creating 762
 - handling 611
 - extraction delay period 181
 - extraction delay start 333, 608
 - extraction environment definition 450
 - designing unit of 325
 - example of 571
 - extraction environment definition file 109, 317, 320, 340
 - handling 610
 - extraction environment definition filename 428
 - extraction error monitoring interval, designing 324
 - extraction information queue file 107, 317, 321, 340
 - handling 611
 - mechanism of data linkage 9, 11
 - modifying organization of 758
 - procedure for storing data in 108
 - size 453
 - extraction information queue filename 452
 - extraction information queue I/O buffer
 - for extraction 324
 - for transmission 327
 - size 453
 - extraction master error information file 108, 317, 321, 340
 - extraction master process 112
 - extraction master status file 108, 317, 321, 340
 - extraction master trace file 109, 322
 - extraction node master error information file 108, 317, 321, 340
 - extraction node master process 112
 - extraction node master trace file 109, 322
 - extraction process 112
 - extraction processing 9, 106
 - by source Datareplicator 106
 - data type subject to 247
 - files used during 107
 - handling of 605
 - organization of processes during 110
 - SQL statement subject to 247
 - table subject to 247
 - timing of 113
 - UAP subject to 247
 - units of 113
 - utility subject to 248
 - extraction processing start method, designing 331
 - extraction processing stop method, designing 333
 - extraction redefinition statement 561
 - extraction restart interval after detecting end of extraction 454
 - extraction server status file 108, 317, 321, 340
 - extraction statement 563
 - extraction status 625
 - extraction system definition 424
 - example of 570
 - extraction system definition file 109, 317, 320, 340
 - handling 610
 - EXTSHM 59, 63
 - extsuppress 453
- F**
- facility for recovering extraction information queue file 1058
 - prerequisites for 1059
 - recovery procedure using 1061
 - FES 1372
 - field name 541, 544, 564
 - to be redefined 562
 - field redefined field name 562
 - file duplexing function, using 726
 - file error, handling during file-duplexed operation 995
 - file organization, examples of 567
 - file transfer program 14
 - file, duplexing 215
 - file_dupenv 441, 504
 - filetype 561
 - FIX attribute table 1372
 - flexible hash partitioning 1372
 - font conventions xiii
 - forced termination 604, 656
 - format update information name 541
 - FREEWORD type 1372

from update information name 480, 545
 front-end server 1372
 functional differences between UNIX and Windows
 Datareplicators 1366

G

GB meaning xvi
 global buffer 1372
 global buffer pool 1372
 glossary 1370
 group import group name 551
 grouped-system switchover 697

H

HA monitor 698
 hash 554
 hash partitioning method 132, 372
 having key range partitioning condition
 statement 553
 having other 554
 HDE_BIN_COL_MAXLEN 59, 85
 notes about 621
 hde_usendcheck(), value returned from 981
 hdechstatus (command syntax) 752
 hdeevent 753
 hdefcopy 755
 hdefstate 756
 hdehost 464
 hdeid 428
 hdemodq 758
 hdenmserv 74
 HDEPATH 58, 84
 hdeprep 762
 hdeserv 69
 hdeservice 463
 hdeshmclean (command syntax) 764
 hdestart 600, 770
 hdestart_n (command syntax) 783
 hdestate 789
 hdestop 603, 799
 hdestop_n (command syntax) 803
 HDS_MST_STDCLOSE 62, 86
 HDS_RFI_ELANG 62, 86
 HDS_RFI_PLANG 62, 86

hds_ubegin()
 editing start instruction function 911
 return values from 913
 hds_ucoleditX(), return values from 961
 hds_ucoleditX, edit column data 952
 hds_ucommon.h 924, 961
 hds_uedit()
 editing and processing instruction
 function 913
 return values from 923
 hds_uend()
 editing termination instruction function 923
 return values from 924
 hds_ureflect.h 924, 961
 hds_usendcheck (edit send data) 973
 hdsagtopt 807
 hdsagtstart 809
 hdsagtstatus 810
 hdsagtstop 812
 hdscnvedt 813
 hdschgstatus (command syntax) 819
 HDSCLTWAITTIME 61, 86
 hdsfcopy 820
 hdsfmkfs 821
 hdsfstate 824
 hdsfstafs 826
 hdsid 492
 hdsMIB 154
 HDSPATH 61, 86
 hdspathlist 155, 829
 hdsrefinfn 831
 hdsrfctl 837
 hdssamqin 841
 hdsservice 494
 hdsshmclean (command syntax) 843
 hdsstart 653, 846
 hdsstate 854
 hdsstop 655, 865
 hdstrcredit 870
 header files, list of 924, 961, 981
 hierarchical system
 application of bidirectional updating system
 to 304

- application of unidirectional updating system to 303
 - designing application to 303
 - high-speed system switchover facility, how to define Datareplicator 707
 - HiRDB 9, 33
 - OSs supported for 1348
 - HiRDB authorization identifier 291
 - HiRDB connection authorization identifier 492
 - HiRDB Dataextractor 2, 3
 - HiRDB Datareplicator 2
 - HiRDB Datareplicator linkage
 - canceling 623, 894
 - checking execution status of 632
 - specifying startup of 411
 - starting 623, 893
 - stopping 623
 - HiRDB Datareplicator linkage facility 411
 - HiRDB definition, needed to use source Datareplicator 485
 - HiRDB facility 1350
 - HiRDB system log file 107
 - HiRDB system, displaying status of 891
 - HiRDB Text Search Plug-in 44
 - HiRDB XML Extension 44
 - HiRDB/Parallel Server 44
 - HiRDB/Single Server 44
 - hirdb_audit_trail
 - extraction system definition 446
 - import system definition 495
 - hirdbds/ 50
 - hirdbusr 492
 - host 1372
 - host name 66, 290, 296, 464
 - registering 90
- I**
- I/O buffers count for reading update information 440
- icon 73
- identifier, example definition of 291
- immediate termination 656
- import 125
 - combining multiple source tables into one table, and then importing to one table 238
 - designing procedure 370
 - service 73
- import command process 129
- import communication master process 129
- import control block name 914
- import data block name 915
- import definition 16, 537
 - server process 129
 - values assumed at omission of 537
- import definition file 127, 366, 368, 390
 - handling 665
- import definition filename 514
- import delay period 181
- import environment definition 509
 - examples of 574, 575
- import environment definition file 127, 366, 368, 390
 - handling 665
- import environment definition filename 494
- import error
 - acquiring untransmitted information 1066
 - information file 390
 - information file size 496
 - skipping 137
- import error information file 366, 369
- import group definition 551
 - grouping by 557
 - when multi-FES facility is used, examples of 583
- import group name 551
- import information editing UOC routine 137, 896
 - creating (UNIX) 900
 - creating (Windows) 908
 - list of header files for 924
 - notes on creating 937
 - samples of 939
 - syntax for functions used with 911
- import information queue file 366, 369, 390
 - handling 666
 - mechanism of data linkage 10, 14
 - overview of import processing 124
 - size 514
 - used in import processing 125
- import information queue filename 513

- import master process 129
 - import master status file 126, 366, 369, 390
 - handling 667
 - import method 130
 - for multi-FES facility 133
 - modifying 837
 - table-based 130, 371
 - transaction-based 130, 371
 - import process 129
 - import information queue file read interval 503, 529
 - import processing 10, 124
 - by target Datareplicator 124
 - commit interval 518
 - conditions for table subject to 250
 - control of 837
 - creating table subject to 250
 - files used during 125
 - handling of 658
 - modifying commit interval of 837
 - organization of processes during 128
 - restarting 837
 - stop event code 518
 - synchronization point processing for 143
 - terminating 837
 - units of 133
 - import processing count reset event code (import environment definition) 518
 - import processing method
 - designing 370
 - designing switching of 388
 - designing when multi-FES facility is used 372
 - import processing start method, designing 381
 - import processing stop method, designing 384
 - import SQL process 129
 - import status file 124, 366, 369, 390
 - size 515
 - import status filename 514
 - import suppress, when table-based import method is used for import processing at target system 1054
 - import suppression list file 128
 - import system definition 489
 - examples of 574, 575
 - import system definition file 127, 366, 368, 390
 - handling 665
 - import table definition 544
 - import trace file 126, 366, 369, 390
 - import transaction 192
 - import transaction synchronization facility 186
 - import UOC process 129
 - independent stop 387
 - index 1373
 - individual back-end servers, definition for 422
 - individual definition section 421, 422
 - info_message_out 436, 498
 - information collection 149
 - executing 159
 - process 154
 - initial start 602, 654
 - initialization procedure
 - at environment configuration stage 595
 - during error recovery 1008
 - in event of inconsistency 1008
 - initializing 595, 677
 - installation
 - Datareplicator (UNIX) 48
 - Datareplicator (Windows) 71
 - int_trc_filesz 438, 502
 - int_trc_getl 525
 - int_trc_getv 455, 471
 - int_trc_lvl 436, 499
 - int_trc_rintvl 439, 502
 - interface area 1373
 - interface block name 912, 914, 923
 - IP address 66
- J**
- JIS8/Shift JIS 263
 - JIS8/Shift JIS code system 433, 497
 - JP1 432, 496, 1373
 - JP1/Cm2 for operations management, using 146
 - JP1/Cm2/Extensible Agent 148
 - JP1/Cm2/Extensible SNMP Agent 45, 148
 - setting up 156
 - JP1/Cm2/Network Node Manager 148
 - JP1/Cm2/Server System Observer 148

K

KB meaning xvi
 keepalive 468, 495
 key column name 477
 key field-name|redefined-field-name 564
 key information block name 915
 key range partitioning 1373
 condition statement 553
 rules for specifying 557
 key range-based partitioning method 132, 133, 371

L

LANG 58, 61
 large files, handling 731
 LDR_CNTRL 59, 64
 linkage obtaining record of updates in chronological order 105
 linkage recovery backup file 1012, 1023, 1373
 literal
 permitted in selection condition 481
 relationship with attribute of selection condition column 482
 load 544
 LOB data 1373
 log-related file, modifying status of 890
 logiosize 453
 logmrg command 1049
 loopback 306
 loopback suppression 306
 when using data linkage facility from XDM/DS 312

M

machine subject to supervision 146
 mainframe database 4, 296
 correspondence between target Datareplicator and 300
 manager 146
 mapping key 16, 250
 designing correspondence of 250
 for PDM2 E2 253
 for RDB1 E2 253
 specification method for ADM 252

specification method for PDM2 E2 253
 specification method for TMS-4V/SP 253
 specification method for XDM/RD E2 252
 specification method for XDM/SD E2 252
 when source database is ADM 252
 when source database is HiRDB 251
 when source database is PDM2 E2 253
 when source database is RDB1 E2 253
 when source database is TMS-4V/SP 253
 when source database is XDM/RD E2 252
 when source database is XDM/SD E2 252

mapping table for converting character codes, editing 813
 mapping_key_check 530
 maxtran 469
 maxtrandata 470
 MB meaning xvi
 memory resource, designing 348, 396
 merge table creation 141
 messages
 descriptive format for 1080
 details about 1085
 list of 1077
 output destination of 1078
 output format of 1078
 output, suppressing 116, 144
 overview of 1078
 selecting English or Japanese language for output of 1082
 MGR 1373
 MIB file 154, 162
 loading 154
 MIB, details about 163
 msglocal 433, 497
 mstservice 434
 multi-FES facility 133
 multi-HiRDB 1373
 mutual system switchover 697, 712
 configuration 697

N

name field name 541
 NETM*Cm2 45
 nocodecncv 543

- node 1374
 - node master process startup service 73
 - node_connection_accept (extraction system definition) 442
 - node_host 448
 - node_pdconfpath 448
 - node_pddir 447
 - node_shlibpath 448
 - node_syslogout (extraction system definition) 445
 - nodecontrol (extraction system definition) 441
 - nodedef (host name) 421
 - NODISCLAIM 59, 64
 - none 478
 - import definition 547
 - normal start 602, 655
 - normal termination 603, 656
 - not_null_unique 478
 - import definition 547
 - note
 - about tables for which COMPRESSED option is specified 675
 - about tables for which WITHOUT ROLLBACK option is specified 674
 - NSAP address 465
 - nsap_address 465
 - nsndidxxx 467
 - null value 1374
- O**
- operating environment definition statement 561
 - operating environment in Windows Vista and Windows Server 2008 92
 - operation, overview of 590
 - operations management
 - organization of processes for 154
 - using NETM*Cm2 146
 - OSI protocol 69, 298
 - overwrite 469
 - overwrite_continue 469
- P**
- partial initial start 602, 655
 - partitioning key index 1374
 - partitioning method
 - hash 132, 372
 - key range-based 132, 133, 371
 - table-based 131, 371
 - password 291, 492
 - PATH 58, 61, 84, 86
 - pd_log_rpl_no_standby_file_opr 486
 - pd_rpl_func_control 28
 - pd_rpl_hdepath 486
 - pd_rpl_init_start 485
 - PDCONFPATH 58, 84
 - pdddef 248
 - PDDIR 58, 61, 84, 86
 - PDHOST 58, 61, 84, 86
 - PDLANG 58, 61
 - pdload 248
 - pdlogchg 890
 - pdl 891
 - PDM2 E2 9, 39
 - PDMII E2 37
 - PDMJANL 127
 - pdmod 248
 - PDNAMEPORT 58, 61, 84, 86
 - pdrbal 248
 - pdrorg 248
 - pdrplstart 893
 - pdrplstop 894
 - pdrstr 248
 - PDUSER 58, 84
 - permitted character conventions xv
 - planned system switchover 697
 - plug-in 1374
 - port number 69, 1374
 - position data start position 562
 - prg_eventno 471
 - process 1374
 - protocol 465
 - protocol1 493
 - protocol2 493
 - PSALLOC 59, 63
- Q**
- queue_read_wait_interval 471
 - queuesize 453, 514
 - qufilexxx 452, 513

qufullwarn 455
 quiosize 453

R

RD-node 1374
 RDB1 E2 9, 39
 RDM2 E2 296
 readbufnum 470
 rebalancing utility 248
 reception 124
 reception process 129
 recover_info_send (extraction system definition) 446
 recover_info_send_interval (extraction system definition) 447
 redefined field
 attribute 562
 field attributes 562
 lengths 562
 ref_data_backspace 522
 ref_wait_interval 503, 529
 refenvxxx 494
 reffile 514
 reflect_counter_reset (import system definition) 506
 reflect_delay_limit_time (import environment definition) 531
 reflect_mode (transmission environment definition) 473
 reflect_trn_max_sqlnum (import environment definition) 534
 reflect_tselector 494
 regular file 1374
 relational database 16
 relational-type database 16
 remote control 152
 executing 160
 repetition column 26, 1375
 designing 280
 replication facility 2, 1375
 reserved words 1362
 resource
 designing 389
 designing (source Datareplicator) 339
 designing (target HiRDB) 380
 resource_chk_err

extraction system definition 446
 import system definition 507

restart 602, 655
 restartmode 516
 restruct 562
 retry_interval (transmission environment definition) 468
 retrynum 468
 return value 913
 row partitioning 1375

S

SAM file 14
 creating file used with target Datareplicator 366
 extracting update information from 841
 file and process used in import processing 127
 file settings used with target Datareplicator 370
 handling 672
 in PDM2 E2 log format 560
 target Datareplicator's disk resources 390
 send data UOC routine 964
 creation procedure (UNIX) 966
 creation procedure (Windows) 970
 notes on creating 981
 samples of 983
 send target identifier 480
 send_counter_reset (extraction system definition) 445
 send_delay_limit_time (transmission environment definition) 472
 sendcontrol 428
 senddefxx 430
 senddefxxxx 431
 sendhdsid 462
 sendidxx 429
 sendidxxxx 430
 sendintvl 465
 sendintvl_scale 466
 sendprocnum 439
 senduoc 465
 server 1375

- service name 69, 290, 295, 298, 463, 494
 - registering 89
 - used for communication with source master process and source node master process 89
 - used for communication with source system 90
 - used for communication with target system 89
- services 73
 - registered during installation 73
- services file 74, 89
 - information specified in 89
- set_tool1 command (data linkage recovery facility) 1052
- set_tool2 command (data linkage recovery facility) 1052
- setup_tool1 command 1051
- setup_tool2 command 1052
- SGMLTEXT type 1375
- shared resource
 - deleting source Datareplicator's (command syntax) 764
 - deleting target Datareplicator's (command syntax) 843
- shiftspace_cnv 521
- SHLIB_PATH 58, 61
- simultaneous start 332, 382, 605, 658
- simultaneous stop 333, 386
- skip_codecnv_error 525
- skip_mvcelmwarn 524
- skip_sqlcode 523
- smt_editbufsize 440
- smt_queue_read_wait_interval 441
- smt_readbufnum 440
- smt_sendintvl 439
- smt_sendintvl_scale 440
- SNMP agent process 154
- SNMP service, setting up 156
- software configuration 32
 - for data linkage system 32
 - for linking data from HiRDB to mainframe database 35
 - for linking data from mainframe database to HiRDB 37
 - for linking data from mainframe database to HiRDB using SAM file 39
 - for linking data from one HiRDB to another 32
- source database 4
 - conditions for 247
 - correspondence between XDM/DS and 297
 - correspondence to target database 16
 - designing correspondence between target database and 247
- source Datareplicator 4
 - changing status of (command syntax) 752
 - collecting status information of 789
 - correspondence between source HiRDB and 287
 - correspondence between target Datareplicator and 288
 - correspondence between XDM/DS and 294
 - designing 313
 - directory structure of 51, 77
 - environment variables for 58, 84
 - error handling procedures for 986
 - file organization of 313
 - handling of files used with 610
 - issuing event at 753
 - operation of 605
 - specifying name of directory used by 411
 - starting 600, 770
 - terminating 600, 799
- source Datareplicator definition 415, 485
 - examples of 569
 - templates for 570
- source Datareplicator identifier 290, 428, 492
- source definition, examples of 576
- source HiRDB
 - correspondence between source Datareplicator and 287
 - designing 411
 - handling of 623
 - handling procedure 632
- source HiRDB database 4
- source HiRDB Datareplicator directory name 486
- source system 4
 - configuration of 33, 36, 38, 40

- correspondence to target system 33, 36, 38, 40
- designing correspondence between target system and 285
- error handling procedure 1002
- space character conversion rule 265, 266
- SQL 30
- SQL error, skipping 141
- SQL process segments count, divide into 556
- sql_lockerr_retrynum (import environment definition) 532
- sqlconvopt1 545
- sqlconvopt2 545
- sqlerr_skip_info 524
- standby-less system switchover (effects distributed) facility, using 715
- start mode
 - for source Datareplicator 602
 - for target Datareplicator 654
- starting
 - source Datareplicator 600
 - target Datareplicator 653
- startmode 515
- statistical activity analysis utility 127
- statistics log file 1375
- statsfile 514
- statssize 515
- status file 108, 126, 1375
 - handling 614, 666
- status information, collecting 113, 136
- status monitoring 148
 - executing 157
 - service 73
- stop event 208
- structured database 16
- structured-type database 16
- superuser 1375
- supervised machine 146
 - operating 162
 - setting up 155, 156
- supervisor machine 146
 - operating 157
 - setting up 154
- swapping 125
- switchover organization-1-to-1 697
- switchover organization-2-to-1 697
- symbol conventions xiii
- syncgroup001 (import system definition) 505
- syncgrp_discintvl (import system definition) 505
- synchronization managing process 129
- synchronization point 1375
- synchronization point dump file 1376
- synchronization point processing for import processing 143
- synchronous event 208
- synchronous import group 195
- synchronous import group name (hdsstate command syntax) 854
- syncterm 434
- syncwait_limit_time (import system definition) 506
- syncwait_limit_tran_count (import system definition) 505
- syslog_message_suppress 432, 496
- syslogfile
 - character encoding conversion 95
 - improving reliability of 95
- syslogout 432, 496
- system call errors, list of 1337
- system common definition 485
- system configuration, examples of 567
- system design 221
- system file 1376
- system log file 9, 1376
 - handling of 624
 - manipulating 628
- system log I/O buffer 322, 323
 - size 453
- system manager 1376
- system RDAREA 1376
- system switchover facility, using 695

T

- T selector 298, 465, 494
 - value 69
- table creation procedure 250
- table identifier 477, 546, 552
- table-based import event 208
- table-based import event code 517

- table-based import method 130, 371
- table-based import restart event 208
- table-based import restart event code 517
- table-based partitioning method 131, 371
- target database 4
 - correspondence between XDM/DS and 296
 - correspondence to source database 16
 - designing correspondence between source database and 247
- target Datareplicator 4
 - changing status of (command syntax) 819
 - collecting status information of 854
 - correspondence between mainframe database and 300
 - correspondence between source Datareplicator and 288
 - correspondence between target HiRDB and 290
 - correspondence between XDM/DS and 297
 - designing 364
 - designing correspondence between files and 364
 - directory structure of 54, 80
 - environment variables for 61, 85
 - error handling procedures for 996
 - example of communications environment for 69
 - handling of files used with 665
 - starting 653, 846
 - terminating 653, 865
- target Datareplicator definition 416
 - examples of 573
 - templates for 573
- target Datareplicator identifier 290
- target definition, examples of 576
- target HiRDB database 4
- target HiRDB, correspondence between target Datareplicator and 290
- target identifier 289, 429, 480, 753, 772, 789, 799
- target system 4
 - configuration of 33, 36, 38, 40
 - correspondence to source system 33, 36, 38, 40
 - designing correspondence between source system and 285
 - error handling procedure at 993
- target system identifier 462
- TB meaning xvi
- tblcheck 520
- tblcmtintvl 520
- TCP/IP 1376
- templates 570, 573
 - /opt/hirdbds/lib/sample/ 570, 573
- Terminal Service, using 90
- terminating
 - source Datareplicator 603
 - target Datareplicator 655
- termination mode
 - for source Datareplicator 603
 - for target Datareplicator 656
- termlevel 435
- time-ordered information
 - available types of 282
 - collecting 135
 - defining table for collection of 283
 - table, creating 282
- time-specified start method 383, 659
- timestamp 545
- TMS-4V/SP 9, 37, 296
- to update information name 477, 564
- transaction 1376
 - management information buffer 326
 - settlement 1000
 - settlement when using two-phase commit method 1000
- transaction branch information 193
- transaction-based import event 208
- transaction-based import event code 517
- transaction-based import method 130, 371
- transaction-based import restart event 208
- transaction-based import restart event code 517
- transmission 107, 113
 - designing procedure 325
- transmission delay period 181
- transmission delay start 332, 606
- transmission environment definition 460
 - designing unit of 331

- examples of 571
- transmission environment definition file 109, 317, 320, 340
 - handling 611
- transmission environment definition filename 430
- transmission interval 328
 - for transmission processing, designing 328
- transmission master process 112, 114
- transmission master process transmission interval 439
- transmission processes 112
 - controlling number of 114
 - extraction information queue file read interval 441, 471
 - to be started, maximum 439
- transmission processing, designing reduced-mode 329
- transmission statement, specification of 479
- transmission suppressed original receiver identifier 467
- transmission target, adding 610
- trigger 1376
- trncmtintvl 519
- tuning 737
- two-phase commit method, checking transaction status while using 999
- TZ 58, 61, 84, 86

U

- UAC 92
- UAP 1377
 - embedded 1372
 - notes on execution 247
- UCS2 263
- ujcodekind 529
- ukey column-name 477
- undefcode_cnv 522
- unextracted data storage file 128, 366, 370, 390
 - handling 672
- unidirectional updating system 302
 - application to hierarchical system 303
- unimported information file 126, 366, 369, 390
 - handling 669
 - size 515

- unimported information filename
 - primary 515
 - secondary 515
- uninstallation 74
 - of Datareplicator 49, 74
- unique 478
 - import definition 547
- unit control information definition 486
- unit identifier 891
- UNIX Datareplicator, functional difference from Windows Datareplicator 1366
- unload log file 1377
- unload statistics log file 1377
- unreffile1 515
- unreffile2 515
- unreffilesz 515
- unset_tool command (data linkage recovery facility) 1053
- unsetup_tool command (command of data linkage recovery facility) 1053
- UOC 116, 137
- UOC routine
 - column data editing 137
 - import information editing 137
 - sending data 116
- update information 9, 16, 1377
 - acquiring over time, record of 244
 - checking 831
 - dataset 11, 14
 - editing buffer 328
 - editing buffer size 440, 470
 - editing buffer, specifying size of 328
 - input process 129
 - processing with user own coding 137
 - to be sent 479
- update information definition 560
- update information definition file 127, 366, 369, 390
 - handling 671
- update information definition filename 841
- update information field definition 540
- update information name 545, 564
- update information names count, maximum 434
- update-SQL file 1067
- update-SQL output facility 1066

upgrading 94
 use_convertlib 502
 User Account Control 92
 user LOB RDAREA 1377
 user own coding 116, 137
 user own coding routine 895
 error handling procedure 995, 1003
 user privileges, checking 48, 71
 user1 69
 UTF-8 263
 utf-8 character code system 497
 utf-8 code system 433
 utility, notes on execution of 248

V

version number conventions xvii
 view table 1377
 volume 1377

W

watchintvl 433
 where clause 480
 Windows Datareplicator, functional difference from
 UNIX Datareplicator 1366
 work table file 1377

X

xa_recovery_retry_count (import environment
 definition) 533
 xa_recovery_retry_interval (import environment
 definition) 534
 XDM/DS
 correspondence between source database
 and 297
 correspondence between source Datareplicator
 and 294
 correspondence between target database
 and 296
 correspondence between target Datareplicator
 and 297
 XDM/RD E2 9, 35, 37, 292, 296
 XDM/SD E2 9, 37, 296
 XDM/XT 3