For UNIX Systems
Scalable Database Server

HiRDB Version 8

Command Reference Part II

3000-6-355(F)

HITACHI

■ Relevant program products

List of program products:

For the HP-UX 11.0, HP-UX 11i, or HP-UX 11i V2 (PA-RISC) operating system:

P-1B62-1182 HiRDB/Single Server Version 8 08-00
P-1B62-1382 HiRDB/Parallel Server Version 8 08-00
P-1B62-1582 HiRDB/Single Server Version 8(64) 08-00
P-1B62-1782 HiRDB/Parallel Server Version 8(64) 08-00
P-1B62-1B82 HiRDB/Run Time Version 8 08-00
P-1B62-1C82 HiRDB/Developer's Kit Version 8 08-00
P-1B62-1D82 HiRDB/Run Time Version 8(64) 08-00
P-1B62-1E82 HiRDB/Developer's Kit Version 8(64) 08-00
P-F1B62-11823 HiRDB Staticizer Option Version 8 08-00
P-F1B62-11825 HiRDB Non Recover Front End Server Version 8 08-00
P-F1B62-11826 HiRDB Advanced High Availability Version 8 08-00
P-F1B62-11827 HiRDB Advanced Partitioning Option Version 8 08-00

For the HP-UX 11i V2 (IPF) operating system:

P-1J62-1582 HiRDB/Single Server Version 8(64) 08-00
P-1J62-1782 HiRDB/Parallel Server Version 8(64) 08-00
P-1J62-1D82 HiRDB/Run Time Version 8(64) 08-00
P-1J62-1E82 HiRDB/Developer's Kit Version 8(64) 08-00
P-F1J62-11823 HiRDB Staticizer Option Version 8 08-00
P-F1J62-11825 HiRDB Non Recover Front End Server Version 8 08-00
P-F1J62-11826 HiRDB Advanced High Availability Version 8 08-00
P-F1J62-11827 HiRDB Advanced Partitioning Option Version 8 08-00

For the Solaris 8, 9 or 10 operating system:

P-9D62-1182 HiRDB/Single Server Version 8 08-00
P-9D62-1382 HiRDB/Parallel Server Version 8 08-00
P-9D62-1582 HiRDB/Single Server Version 8(64) 08-00
P-9D62-1782 HiRDB/Parallel Server Version 8(64) 08-00
P-9D62-1B82 HiRDB/Run Time Version 8 08-00
P-9D62-1C82 HiRDB/Developer's Kit Version 8 08-00
P-9D62-1D82 HiRDB/Run Time Version 8(64) 08-00
P-9D62-1E82 HiRDB/Developer's Kit Version 8(64) 08-00
P-F9D62-11823 HiRDB Staticizer Option Version 8 08-00
P-F9D62-11825 HiRDB Non Recover Front End Server Version 8 08-00
P-F9D62-11826 HiRDB Advanced High Availability Version 8 08-00
P-F9D62-11827 HiRDB Advanced Partitioning Option Version 8 08-00

For the AIX(R) 5L V5.1, V5.2 or V5.3 operating system:

P-1M62-1182 HiRDB/Single Server Version 8 08-00
P-1M62-1382 HiRDB/Parallel Server Version 8 08-00
P-1M62-1582 HiRDB/Single Server Version 8(64) 08-00
P-1M62-1782 HiRDB/Parallel Server Version 8(64) 08-00
P-1M62-1B82 HiRDB/Run Time Version 8 08-00
P-1M62-1C82 HiRDB/Developer's Kit Version 8 08-00

P-1M62-1D82 HiRDB/Run Time Version 8(64) 08-00

P-1M62-1E82 HiRDB/Developer's Kit Version 8(64) 08-00

P-F1M62-11823 HiRDB Staticizer Option Version 8 08-00

P-F1M62-11825 HiRDB Non Recover Front End Server Version 8 08-00

P-F1M62-11826 HiRDB Advanced High Availability Version 8 08-00

P-F1M62-11827 HiRDB Advanced Partitioning Option Version 8 08-00

For the Red Hat Linux 7.1, Red Hat Linux 7.2, 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 AS 4 (x86), Red Hat Enterprise Linux ES 4 (x86), Red Hat Enterprise Linux AS 3 (AMD64 & Intel EM64T),* Red Hat Enterprise Linux AS 4 (AMD64 & Intel EM64T), or Red Hat Enterprise Linux ES 4 (AMD64 & Intel EM64T) operating system:

P-9S62-1182 HiRDB/Single Server Version 8 08-00

P-9S62-1382 HiRDB/Parallel Server Version 8 08-00

P-9S62-1B82 HiRDB/Run Time Version 8 08-00

P-9S62-1C82 HiRDB/Developer's Kit Version 8 08-00

P-F9S62-11823 HiRDB Staticizer Option Version 8 08-00

P-F9S62-11825 HiRDB Non Recover Front End Server Version 8 08-00

P-F9S62-11826 HiRDB Advanced High Availability Version 8 08-00

P-F9S62-11827 HiRDB Advanced Partitioning Option Version 8 08-00

* Only operating systems that run on the Intel EM64T are supported.

For the Red Hat Enterprise Linux AS 3 (AMD64 & Intel EM64T),* Red Hat Enterprise Linux AS 4 (AMD64 & Intel EM64T), or Red Hat Enterprise Linux ES 4 (AMD64 & Intel EM64T) operating system:

P-9W62-1182   HiRDB/Single Server Version 8 08-00

P-9W62-1382 HiRDB/Parallel Server Version 8 08-00

P-9W62-1B82 HiRDB/Run Time Version 8 08-00

P-9W62-1C82 HiRDB/Developer's Kit Version 8 08-00

* Only operating systems that run on the Intel EM64T are supported.

For the Red Hat Enterprise Linux AS 3 (IPF) or Red Hat Enterprise Linux AS 4 (IPF) operating system:

P-9V62-1182 HiRDB/Single Server Version 8 08-00

P-9V62-1382 HiRDB/Parallel Server Version 8 08-00

P-9V62-1B82 HiRDB/Run Time Version 8 08-00

P-9V62-1C82 HiRDB/Developer's Kit Version 8 08-00

P-F9V62-11823 HiRDB Staticizer Option Version 8 08-00

P-F9V62-11825 HiRDB Non Recover Front End Server Version 8 08-00

P-F9V62-11826 HiRDB Advanced High Availability Version 8 08-00

P-F9V62-11827 HiRDB Advanced Partitioning Option Version 8 08-00

This edition of the manual is released for the preceding program products, which have been developed under a quality management system that has been certified to comply with ISO9001 and TickIT. This manual may also apply to other program products; for details, see *Software Information* or *Before Installing*.

■ Trademarks

■ Edition history

Edition 1 (3000-6-355(E)): March, 2007

Preface

This manual describes the syntax of the commands used with HiRDB Scalable Database Server HiRDB Version 8.

Intended readers

This manual is intended for users who will be constructing or operating *HiRDB Version* 8 ("HiRDB") relational database systems.

It is assumed that readers of this manual have the following:

- A basic knowledge of managing UNIX or Linux systems
- A basic knowledge of SQL

The following manuals should be read before reading this manual:

- *HiRDB Version* 8 *Installation and Design Guide*
- *HiRDB Version* 8 *System Operation Guide*

Organization of this manual

This manual consists of the following 18 chapters and appendixes:

Chapter *1. Overview of Commands*

Chapter 1 explains the methods of entering commands and the descriptive format for commands.

Chapter *2. Operation Commands*

Chapter 2 explains the operation commands that can be used with HiRDB.

Chapter *3. Database Initialization Utility (pdinit)*

Chapter 3 describes the database initialization utility that defines a physical structure that, in turn, permits files to be used as a HiRDB database.

Chapter *4. Database Definition Utility (pddef)*

Chapter 4 describes the database definition utility that permits modifications of schema definitions and contents.

Chapter *5. Database Load Utility (pdload)*

Chapter 5 describes the database load utility that stores user-provided data in a table.

Chapter *6. Interactive SQL Execution Utility (pdsql)*

Chapter 6 explains the interactive SQL execution utility that can interactively execute SQL statements.

Chapter *7. Database Structure Modification Utility (pdmod)*

Chapter 7 explains the database structure modification utility that can modify the physical structure of a database, such as by adding an RDAREA, expanding the database, or reinitializing the database.

Chapter *8. Database Reorganization Utility (pdrorg)*

Chapter 8 explains the database reorganization utility for maintaining tables and indexes.

Chapter *9. Dictionary Import/Export Utility (pdexp)*

Chapter 9 explains the dictionary import/export utility that migrates table definition information and stored procedure information between HiRDB systems.

Chapter *10. Rebalancing Utility (pdrbal)*

Chapter 10 explains the rebalancing utility that rebalances a table partitioned by the hash partitioning method.

Chapter *11. Free Page Release Utility (pdreclaim)*

Chapter 11 explains the free page release utility that enables you to release free pages (used free pages) during online operation.

Chapter *12. Global Buffer Residence Utility (pdpgbfon)*

Chapter 12 explains the global buffer residence utility that reads table data pages and index pages into a global buffer in advance, such as immediately following HiRDB startup or before starting online jobs.

Chapter *13. Integrity Check Utility (pdconstck)*

Chapter 13 explains the integrity check utility (`pdconstck`), which performs integrity checking and manipulates (sets or releases) the check pending status on tables for which referential constraints or check constraints have been defined.

Chapter *14. Statistics Analysis Utility (pdstedit)*

Chapter 14 explains the statistics analysis utility that edits statistical information, such as information about HiRDB system activities.

Chapter *15. Database Condition Analysis Utility (pddbst)*

Chapter 15 explains the database condition analysis utility, which analyzes the relationship between data dictionary tables or user RDAREAs and the storage status of the index, and summarizes and displays the results of the analysis.

Chapter *16. Optimizing Information Collection Utility (pdgetcst)*

Chapter 16 explains the optimizing information collection utility, which collects information for optimizing based on cost and stores the information in the dictionary table.

Chapter *17. Access Path Display Utility (pdvwopt)*

Chapter 17 explains the access path display utility, which displays access path information that was determined by the SQL optimization processing.

Chapter *18. Database Copy Utility (pdcopy)*

Chapter 18 explains how to use the database copy utility (`pdcopy`), which makes a backup of a database.

Chapter *19. Database Recovery Utility (pdrstr)*

Chapter 19 explains how to use the database recovery utility, which is used to recover databases and re-create log point information files.

Chapter *20. Registry Facility Initialization Utility (pdreginit)*

Chapter 20 explains the registry facility initialization utility, which can be used to register the registry facility for using plug-in utilities.

Appendix *A. List of Commands*

Appendix A provides a list of operation commands and utilities.

Appendix *B. Lock Mode During Command Execution*

Appendix B describes the resources that are locked while an operation command or utility is executing.

Appendix *C. RDAREA Status During Command Execution*

Appendix C describes the RDAREA statuses during execution of commands.

Appendix *D. Maximum Number of Concurrently Executable Utilities*

Appendix D describes the maximum number of utilities that can be executed concurrently.

Appendix *E. Creation of Input Data Files for the Database Load Utility*

Appendix E explains the creation of data files that can be input to the database load utility (`pdload`).

Appendix *F. Creation of a UOC for Use by pdload and pdrorg*

Appendix F explains the creation of a UOC for use by `pdload` and `pdrorg`.

Appendix *G. Number of Concurrent Command Connections*

Appendix G explains the number of concurrent command connections.

Appendix *H. List of Command Return Codes*

Appendix H explains the return codes from commands.

Related publications

This manual is related to the following manuals, which should be read as required.

HiRDB (for UNIX)

- *For UNIX Systems HiRDB Version 8 Description* (3000-6-351(E))
- *For UNIX Systems HiRDB Version 8 Installation and Design Guide* (3000-6-352(E))
- *For UNIX Systems HiRDB Version 8 System Definition* (3000-6-353(E))
- *For UNIX Systems HiRDB Version 8 System Operation Guide* (3000-6-354(E))
- *HiRDB Staticizer Option Version 7 Description and User's Guide* (3000-6-282(E))
- *For UNIX Systems HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide* (3000-6-364)*

HiRDB (for Windows)

- *For Windows Systems HiRDB Version 8 Description* (3020-6-351(E))
- *For Windows Systems HiRDB Version 8 Installation and Design Guide* (3020-6-352(E))
- *For Windows Systems HiRDB Version 8 System Definition* (3020-6-353(E))
- *For Windows Systems HiRDB Version 8 System Operation Guide* (3020-6-354(E))
- *For Windows Systems HiRDB Version 8 Command Reference* (3020-6-355(E))

HiRDB (for both Windows and UNIX)

- *HiRDB Version 8 UAP Development Guide* (3020-6-356(E))
- *HiRDB Version 8 SQL Reference* (3020-6-357(E))
- *HiRDB Version 8 Messages* (3020-6-358(E))
- *HiRDB Datareplicator Version 8 Description, User's Guide and Operator's Guide* (3020-6-360(E))
- *HiRDB Dataextractor Version 8 Description, User's Guide and Operator's Guide* (3020-6-362(E))

* This manual has been published in Japanese only; it is not available in English.

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

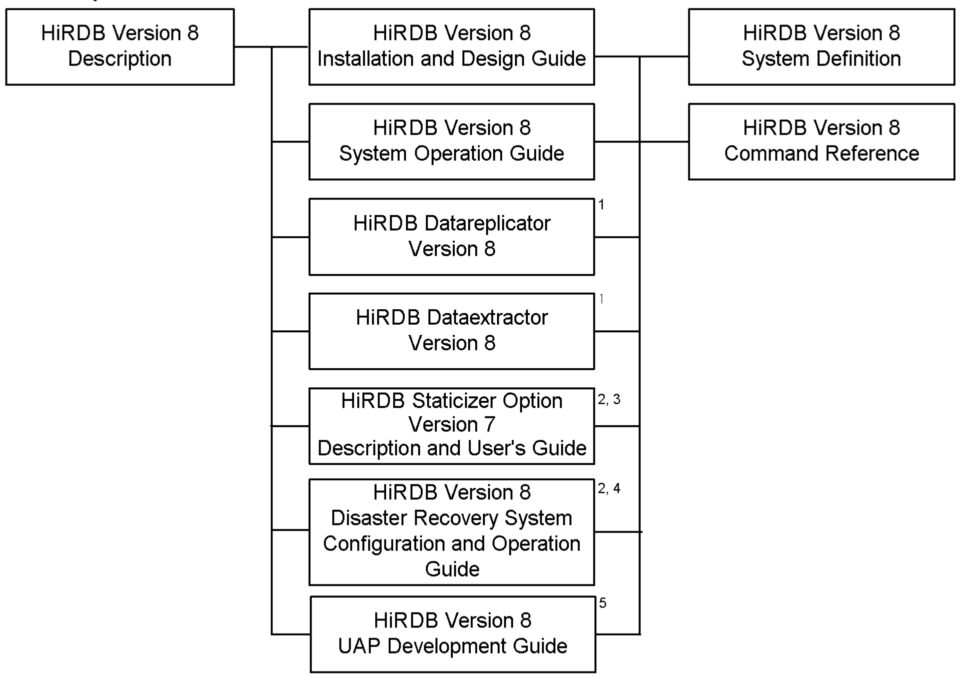
Others

- *HiRDB External Data Access Version 7 Description and User's Guide* (3000-6-284(E))
- *JP1 Version 6 JP1/VERITAS NetBackup v4.5 Agent for HiRDB License Description and User's Guide* (3020-3-D79(E))

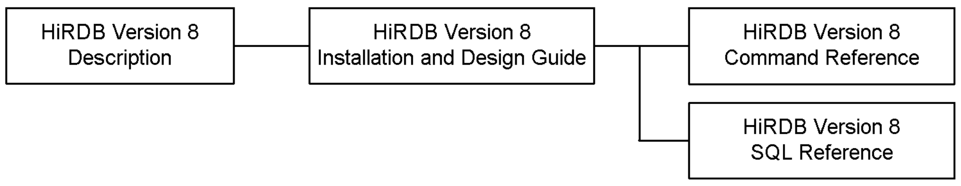
Organization of HiRDB manuals

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.

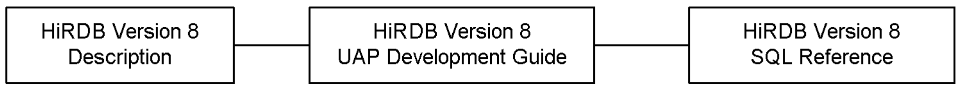
Manuals for system administrators:


HiRDB Version 8 Description
HiRDB Version 8 Installation and Design Guide
HiRDB Version 8 System Definition
HiRDB Version 8 System Operation Guide
HiRDB Version 8 Command Reference
HiRDB Datareplicator Version 8 1
HiRDB Dataextractor Version 8 1
HiRDB Staticizer Option Version 7 Description and User's Guide 2, 3
HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide 2, 4
HiRDB Version 8 UAP Development Guide 5

Manuals for users who create tables:


HiRDB Version 8 Description
HiRDB Version 8 Installation and Design Guide
HiRDB Version 8 Command Reference
HiRDB Version 8 SQL Reference

Manuals for users who create or execute UAPs:


HiRDB Version 8 Description
HiRDB Version 8 UAP Development Guide
HiRDB Version 8 SQL Reference

[1] Read if you use the replication facility to link data.
[2] Published for UNIX only. There is no corresponding Windows manual.
[3] Read if you use the inner replica facility.
[4] Read if you are configuring a disaster recovery system.
[5] Must be read if you are linking HiRDB to an OLTP system.

Conventions: Abbreviations

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

Name of product or other entity	Representation	
HiRDB/Single Server Version 8	HiRDB/Single Server	HiRDB or HiRDB Server
HiRDB/Single Server Version 8(64)		
HiRDB/Parallel Server Version 8	HiRDB/Parallel Server	
HiRDB/Parallel Server Version 8(64)		
HiRDB/Developer's Kit Version 8	HiRDB/ Developer's Kit	HiRDB Client
HiRDB/Developer's Kit Version 8(64)		
HiRDB/Run Time Version 8	HiRDB/Run Time	
HiRDB/Run Time Version 8(64)		
HiRDB Datareplicator Version 7	HiRDB Datareplicator	
HiRDB Dataextractor Version 7	HiRDB Dataextractor	
HiRDB Text Search Plug-in Version 7	HiRDB Text Search Plug-in	
HiRDB Spatial Search Plug-in Version 3	HiRDB Spatial Search Plug-in	
HiRDB Staticizer Option Version 8	HiRDB Staticizer Option	
HiRDB LDAP Option Version 8	HiRDB LDAP Option	
HiRDB Advanced Partitioning Option Version 8	HiRDB Advanced Partitioning Option	
HiRDB Advanced High Availability Version 8	HiRDB Advanced High Availability	
HiRDB Non Recover Front End Server Version 8	HiRDB Non Recover FES	
HiRDB Disaster Recovery Light Edition Version 8	HiRDB Disaster Recovery Light Edition	
HiRDB External Data Access Version 8	HiRDB External Data Access	
HiRDB External Data Access Adapter Version 8	HiRDB External Data Access Adapter	
HiRDB Adapter for XML - Standard Edition	HiRDB Adapter for XML	
HiRDB Adapter for XML - Enterprise Edition		
HiRDB Control Manager	HiRDB CM	
HiRDB Control Manager Agent	HiRDB CM Agent	

Name of product or other entity	Representation
Hitachi TrueCopy	TrueCopy
Hitachi TrueCopy basic	
TrueCopy	
TrueCopy remote replicator	
JP1/Automatic Job Management System 2	JP1/AJS2
JP1/Automatic Job Management System 2 - Scenario Operation	JP1/AJS2-SO
JP1/Cm2/Extensible SNMP Agent	JP1/ESA
JP1/Cm2/Extensible SNMP Agent for Mib Runtime	
JP1/Cm2/Network Node Manager	JP1/NNM
JP1/Integrated Management - Manager	JP1/Integrated Management or JP1/IM
JP1/Integrated Management - View	
JP1/Magnetic Tape Access	EasyMT
EasyMT	
JP1/Magnetic Tape Library	MTguide
JP1/NETM/DM	JP1/NETM/DM
JP1/NETM/DM Manager	
JP1/Performance Management	JP1/PFM
JP1/Performance Management Agent for HiRDB	JP1/PFM-Agent for HiRDB
JP1/Performance Management - Agent for Platform	JP1/PFM-Agent for Platform
JP1/Performance Management/SNMP System Observer	JP1/SSO
JP1/VERITAS NetBackup BS v4.5	NetBackup
JP1/VERITAS NetBackup v4.5	
JP1/VERITAS NetBackup BS V4.5 Agent for HiRDB License	JP1/VERITAS NetBackup Agent for HiRDB License
JP1/VERITAS NetBackup V4.5 Agent for HiRDB License	
JP1/VERITAS NetBackup 5 Agent for HiRDB License	
OpenTP1/Server Base Enterprise Option	TP1/EE

Name of product or other entity	Representation	
Virtual-storage Operating System 3/Forefront System Product	VOS3/FS	VOS3
Virtual-storage Operating System 3/Leading System Product	VOS3/LS	
Extensible Data Manager/Base Extended Version 2 XDM basic program XDM/BASE E2	XDM/BASE E2	
XDM/Data Communication and Control Manager 3 XDM Data communication control XDM/DCCM3	XDM/DCCM3	
XDM/Relational Database XDM/RD	XDM/RD	XDM/RD
XDM/Relational Database Extended Version 2 XDM/RD E2	XDM/RD E2	
VOS3 Database Connection Server	DB Connection Server	
DB2 Universal Database for OS/390 Version 6	DB2	
^{DNCWARE} ClusterPerfect (Linux Version)	ClusterPerfect	
Microsoft$_{(R)}$ Excel	Microsoft Excel or Excel	
Microsoft$_{(R)}$ Visual C++$_{(R)}$	Visual C++ or C++	
Oracle 8i	ORACLE	
Oracle 9i		
Oracle 10g		
Sun Java^{TM} System Directory Server	Sun Java System Directory Server or Directory Server	
HP-UX 11i V2 (IPF)	HP-UX or HP-UX (IPF)	
Red Hat Linux	Linux	
Red Hat Enterprise Linux		
Red Hat Enterprise Linux AS 3 (IPF)	Linux (IPF)	Linux
Red Hat Enterprise Linux AS 4 (IPF)		
Red Hat Enterprise Linux AS 3(AMD64 & Intel EM64T)	Linux (EM64T)	
Red Hat Enterprise Linux AS 4(AMD64 & Intel EM64T)		
Red Hat Enterprise Linux ES 4(AMD64 & Intel EM64T)		
turbolinux 7 Server for AP8000	Linux for AP8000	

Name of product or other entity	Representation	
Microsoft$_{(R)}$ Windows NT$_{(R)}$ Workstation Operating System Version 4.0	Windows NT	
Microsoft$_{(R)}$ Windows NT$_{(R)}$ Server Network Operating System Version 4.0		
Microsoft$_{(R)}$ Windows$_{(R)}$ 2000 Professional Operating System	Windows 2000	
Microsoft$_{(R)}$ Windows$_{(R)}$ 2000 Server Operating System		
Microsoft$_{(R)}$ Windows$_{(R)}$ 2000 Datacenter Server Operating System		
Microsoft$_{(R)}$ Windows$_{(R)}$ 2000 Advanced Server Operating System	Windows 2000 or Windows 2000 Advanced Server	
Microsoft$_{(R)}$ Windows Server^{TM} 2003, Standard Edition	Windows Server 2003	
Microsoft$_{(R)}$ Windows Server^{TM} 2003, Enterprise Edition		
Microsoft$_{(R)}$ Windows Server^{TM} 2003 R2, Standard Edition	Windows Server 2003 R2 or Windows Server 2003	
Microsoft$_{(R)}$ Windows Server^{TM} 2003 R2, Enterprise Edition		
64 bit Version Microsoft$_{(R)}$ Windows Server^{TM} 2003, Enterprise Edition (IPF)	Windows Server 2003 (IPF) or Windows Server 2003	
Microsoft$_{(R)}$ Windows Server^{TM} 2003, Standard x64 Edition	Windows Server 2003 or Windows Server 2003 x64 Editions	Windows (x64)
Microsoft$_{(R)}$ Windows Server^{TM} 2003, Enterprise x64 Edition		
Microsoft$_{(R)}$ Windows Server^{TM} 2003 R2, Standard x64 Edition	Windows Server 2003, Windows Server 2003 R2 or Windows Server 2003 x64 Editions	
Microsoft$_{(R)}$ Windows Server^{TM} 2003 R2, Enterprise x64 Edition		
Microsoft$_{(R)}$ Windows$_{(R)}$ XP Professional x64 Edition	Windows XP or Windows XP x64 Edition	
Microsoft$_{(R)}$ Windows$_{(R)}$ XP Professional Operating System	Windows XP Professional	Windows XP

Name of product or other entity	Representation	
Microsoft$_{(R)}$ Windows$_{(R)}$ XP Home Edition Operating System	Windows XP Home Edition	
Single server	SDS	
System manager	MGR	
Front-end server	FES	
Dictionary server	DS	
Back-end server	BES	

- Windows 2000, Windows XP, and Windows Server 2003 may be referred to collectively as *Windows*.
- The HiRDB directory path is represented as `$PDDIR`.
- The hosts file means the `hosts` file stipulated by TCP/IP (including the `/etc/hosts` file).

This manual also uses the following abbreviations:

Abbreviation	Full name or meaning
ACK	Acknowledgement
ADM	Adaptable Data Manager
ADO	ActiveX Data Objects
ADT	Abstract Data Type
AP	Application Program
API	Application Programming Interface
ASN.1	Abstract Syntax Notation One
BES	Back End Server
BLOB	Binary Large Object
BOM	Byte Order Mark
CD-ROM	Compact Disc - Read Only Memory
CGI	Common Gateway Interface
CLOB	Character Large Object
CMT	Cassette Magnetic Tape

Abbreviation	Full name or meaning
COBOL	Common Business Oriented Language
CORBA(R)	Common ORB Architecture
CPU	Central Processing Unit
CSV	Comma Separated Values
DAO	Data Access Object
DAT	Digital Audio Taperecorder
DB	Database
DBM	Database Module
DBMS	Database Management System
DDL	Data Definition Language
DF for Windows NT	Distributing Facility for Windows NT
DF/UX	Distributing Facility/for UNIX
DIC	Dictionary Server
DLT	Digital Linear Tape
DML	Data Manipulate Language
DNS	Domain Name System
DOM	Document Object Model
DS	Dictionary Server
DTD	Document Type Definition
DTP	Distributed Transaction Processing
DWH	Data Warehouse
EUC	Extended UNIX Code
EX	Exclusive
FAT	File Allocation Table
FD	Floppy Disk
FES	Front End Server
FQDN	Fully Qualified Domain Name

Abbreviation	Full name or meaning
FTP	File Transfer Protocol
GUI	Graphical User Interface
HBA	Host Bus Adapter
HD	Hard Disk
HTML	Hyper Text Markup Language
ID	Identification number
IP	Internet Protocol
IPF	Itanium(R) Processor Family
JAR	Java Archive File
Java VM	Java Virtual Machine
JDBC	Java Database Connectivity
JDK	Java Developer's Kit
JFS	Journaled File System
JFS2	Enhanced Journaled File System
JIS	Japanese Industrial Standard code
JP1	Job Management Partner 1
JRE	Java Runtime Environment
JTA	Java Transaction API
JTS	Java Transaction Service
KEIS	Kanji processing Extended Information System
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
LIP	loop initialization process
LOB	Large Object
LRU	Least Recently Used
LTO	Linear Tape-Open
LU	Logical Unit

Abbreviation	Full name or meaning
LUN	Logical Unit Number
LVM	Logical Volume Manager
MGR	System Manager
MIB	Management Information Base
MRCF	Multiple RAID Coupling Feature
MSCS	Microsoft Cluster Server
NAFO	Network Adapter Fail Over
NAPT	Network Address Port Translation
NAT	Network Address Translation
NIC	Network Interface Card
NIS	Network Information Service
NTFS	New Technology File System
ODBC	Open Database Connectivity
OLAP	Online Analytical Processing
OLE	Object Linking and Embedding
OLTP	On-Line Transaction Processing
OOCOBOL	Object Oriented COBOL
ORB	Object Request Broker
OS	Operating System
OSI	Open Systems Interconnection
OTS	Object Transaction Service
PC	Personal Computer
PDM II E2	Practical Data Manager II Extended Version 2
PIC	Plug-in Code
PNM	Public Network Management
POSIX	Portable Operating System Interface for UNIX
PP	Program Product

Abbreviation	Full name or meaning
PR	Protected Retrieve
PU	Protected Update
RAID	Redundant Arrays of Inexpensive Disk
RD	Relational Database
RDB	Relational Database
RDB1	Relational Database Manager 1
RDB1 E2	Relational Database Manager 1 Extended Version 2
RDO	Remote Data Objects
RiSe	Real time SAN replication
RM	Resource Manager
RMM	Resource Manager Monitor
RPC	Remote Procedure Call
SAX	Simple API for XML
SDS	Single Database Server
SGML	Standard Generalized Markup Language
SJIS	Shift JIS
SNMP	Simple Network Management Protocol
SQL	Structured Query Language
SQL/K	Structured Query Language / VOS K
SR	Shared Retrieve
SU	Shared Update
TCP/IP	Transmission Control Protocol / Internet Protocol
TM	Transaction Manager
TMS-4V/SP	Transaction Management System - 4V / System Product
UAP	User Application Program
UOC	User Own Coding
VOS K	Virtual-storage Operating System Kindness

Abbreviation	Full name or meaning
VOS1	Virtual-storage Operating System 1
VOS3	Virtual-storage Operating System 3
WS	Workstation
WWW	World Wide Web
XDM/BASE E2	Extensible Data Manager / Base Extended Version 2
XDM/DF	Extensible Data Manager / Distributing Facility
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
XFIT	Extended File Transmission program
XML	Extensible Markup Language

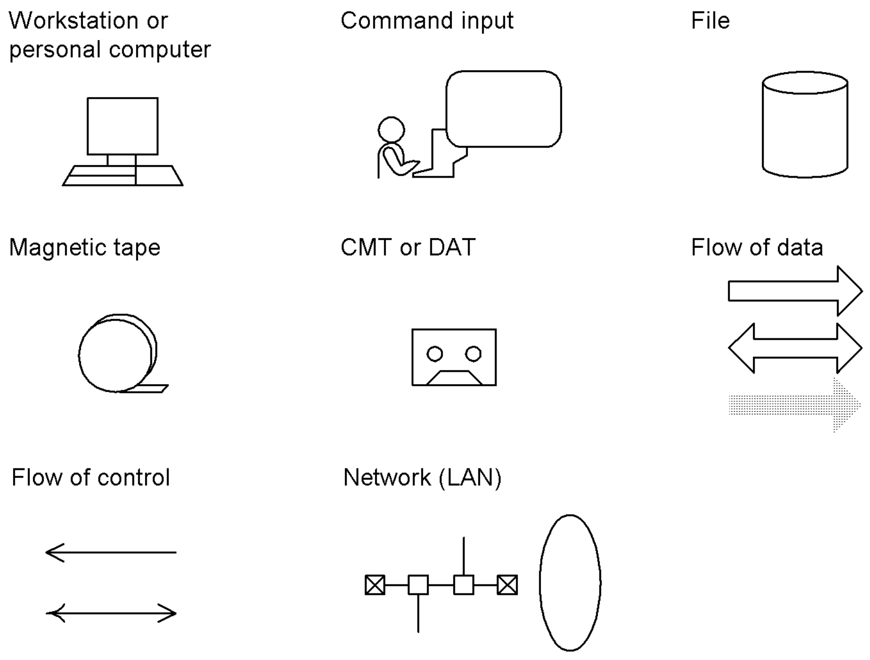
Log representations

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 5L	`/var/adm/ras/syslog`
Linux	`/var/log/messages`

Conventions: Diagrams

This manual uses the following conventions in diagrams:



Conventions: Fonts and symbols

Font and symbol conventions are classified as:

- General font conventions
- Conventions in syntax explanations

These conventions are described below.

General font conventions

The following table lists the general font conventions:

Font	Convention
Bold	Bold type indicates text on a window, other than the window title. Such text includes menus, menu options, buttons, radio box options, or explanatory labels. For example, bold is used in sentences such as the following: • From the **File** menu, choose **Open**. • Click the **Cancel** button. • In the **Enter name** entry box, type your name.

Font	Convention
Italics	Italics are used to indicate a placeholder for some actual text provided by the user or system. Italics are also used for emphasis. For example: • Write the command as follows: `copy` *source-file target-file* • Do *not* delete the configuration file.
`Code font`	A code font indicates text that the user enters without change, or text (such as messages) output by the system. For example: • At the prompt, enter `dir`. • Use the `send` command to send mail. • The following message is displayed: `The password is incorrect.`

Examples of coding and messages appear as follows (although there may be some exceptions, such as when coding is included in a diagram):

```
MakeDatabase
...
StoreDatabase temp DB32
```

In examples of coding, an ellipsis (...) indicates that one or more lines of coding are not shown for purposes of brevity.

Conventions in syntax explanations

Syntax definitions appear as follows:

S`tore`**D**`atabase [temp|`<u>`perm`</u>`]` (*database-name* `...`)

The following table lists the conventions used in syntax explanations. The syntactical characters described below are used to provide a clear explanation of code syntax; you do not actually enter these characters.

Example font or symbol	Convention
`StoreDatabase`	Code-font characters must be entered exactly as shown.
database-name	This font style marks a placeholder that indicates where appropriate characters are to be entered in an actual command.
`SD`	Bold code-font characters indicate the abbreviation for a command.
<u>`perm`</u>	Underlined characters indicate the default value.
`[ ]`	Square brackets enclose an item or set of items whose specification is optional.
|	Only one of the options separated by a vertical bar can be specified at the same time.

Example font or symbol	Convention
...	An ellipsis (...) indicates that the item or items enclosed in ( ) or [ ] immediately preceding the ellipsis may be specified as many times as necessary.
()	Parentheses indicate the range of items to which the vertical bar (|) or ellipsis (...) is applicable.
~	The user-specified value preceding the swung dash must be specified in accordance with the attributes following the swung dash.
<< >>	Double angle brackets enclose the default value assumed by the system when the specification is omitted.
< >	Angle brackets enclose the syntax element notation for a user-specified value.
(( ))	Double parentheses enclose the permitted range of values that can be specified.

Syntax element notations

Syntax element notations explain the types of user-specified values.

Syntax element notation	Explanation
<*alphabetic*>	Alphabetic characters (A-Z, a-z) and _ (underline)
<*alphabetic symbol*>	Alphabetic characters (A-Z, a-z), #, @, \
<*alphanumeric*>	Alphabetic characters and numeric characters (0-9)
<*alphanumeric symbol*>	Alphabetic symbols and numeric characters
<*unsigned integer*>	Numeric characters
<*hexadecimal*>	Numeric characters and (A-F, a-f)
<*identifier*>[1]	Alphanumeric character string beginning with an alphabetic character
<*symbolic name*>	Alphanumeric symbol string beginning with an alphabetic symbol
<*character string*>	String of any characters
<*path name*>[2]	Includes symbolic names, forward slashes (/), and periods (.).
<*HiRDB file name*>	Character string consisting of alphabetic characters (A-Z, a-z), numeric characters (0-9), . (period), _ (underline), and @ (maximum 30 characters)

Note

Alphabetic characters are case sensitive (i.e., lowercase alphabetic characters are distinguished from uppercase alphabetic characters).

[1] An RDAREA name is an alphanumeric character string beginning with an alphabetic

character or special character, and can include alphanumeric characters, underscores (_), and spaces. If an RDAREA name includes a space, the entire name must be enclosed in double quotation marks (`"`).

A host name is a character string that can include alphabetic characters (`A` to `Z`, `a` to `z`), numeric characters, periods (`.`), hyphens (`-`), and underscores (_). A host name can begin with a numeric character.

2 Path names depend on the operating system being used. The backslash (\) must not be used in a HiRDB file system area name.

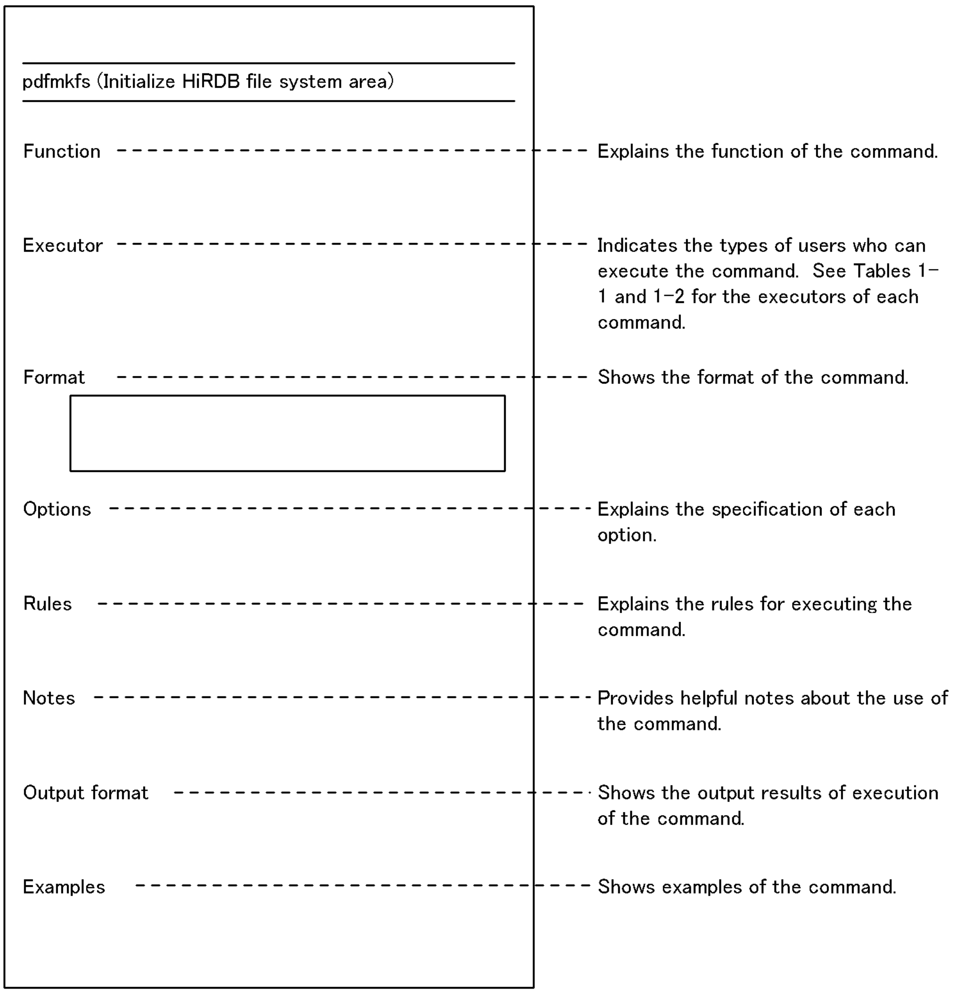
Notations used in formulas

The following notations are used in computational expressions:

Notation	Explanation
↑ ↑	Round up the result to the next integer. Example: The result of ↑ 34 ÷ 3 ↑ is 12.
↓ ↓	Discard digits following the decimal point. Example: The result of ↓ 34 ÷ 3 ↓ is 11.
MAX	Select the largest value as the result. Example: The result of max(10, 2 × 4, 3 + 8) is 11.
MIN	Select the smallest value as the result. Example: The result of min(10, 2 × 4, 3 + 8) is 8.

Conventions in command explanations

This manual explains each command in the format shown as follows; note that this format may differ slightly from one command to another:


pdfmkfs (Initialize HiRDB file system area)
Function
Explains the function of the command.
Executor
Indicates the types of users who can execute the command. See Tables 1-1 and 1-2 for the executors of each command.
Format
Shows the format of the command.
Options
Explains the specification of each option.
Rules
Explains the rules for executing the command.
Notes
Provides helpful notes about the use of the command.
Output format
Shows the output results of execution of the command.
Examples
Shows examples of the command.

Conventions: KB, MB, GB, and TB

This manual uses the following conventions:

- 1 KB (kilobyte) is 1,024 bytes.
- 1 MB (megabyte) is $1{,}024^2$ bytes.
- 1 GB (gigabyte) is $1{,}024^3$ bytes.
- 1 TB (terabyte) is $1{,}024^4$ bytes.

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 (when described for the Windows version of HiRDB)
- Updatable online reorganization (when described for the Windows version of HiRDB)
- Sun Java System Directory Server linkage facility
- Simple setup tool

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

- HiRDB Control Manager
- HiRDB Disaster Recovery Light Edition
- HiRDB External Data Access
- HiRDB LDAP Option

Notes on printed manuals

Please note that even though the printed manuals are separated into Part I and Part II, the chapters and page numbers sequentially continue from Part I to Part II. Also, please note that the index is only included in Part II.

Contents

List of figures

List of tables

Chapter

8. Database Reorganization Utility (pdrorg)

This chapter explains the database reorganization utility (`pdrorg`) for maintaining tables and indexes.

This chapter contains the following sections:

8.1 Overview

8.1.1 Functions of pdrorg

The database reorganization utility (`pdrorg`) is used to maintain tables and indexes. Table 8-1 lists the use and functions of `pdrorg`, and Figure 8-1 shows an overview of the functions of `pdrorg`.

Table 8-1: Use and functions of pdrorg

Classification	**Use**	**pdrorg function**						
		A	**B**	**C**	**D**	**E**	**F**	**G**
Table	Improving table storage efficiency (use `pddbst` to check)	Y	—	—	Y	—	—	—
	Message `KFPH00212-I` or `KFPH22017-I` issued	Y	—	—	Y	—	—	—
	Expected search performance not obtained	Y	—	—	Y	—	—	—
	Migrating data to another table	—	Y	Y	—	—	—	—
	Modifying table partitioning conditions	—	Y	Y	—	—	—	—
	Using unload data for `pdload`'s input data file or with a UAP	—	Y	—	—	—	—	—
	Reorganizing tables for unloading purposes separately from reloading purposes	—	Y	Y	—	—	—	—

Classification	Use	pdrorg function						
		A	B	C	D	E	F	G
Index	Specifying `-i n` during execution of `pdload` or `pdrorg`	—	—	—	—	Y	—	—
	Loading data in units of RDAREAs using `pdload` on a table with non-partitioned indexes, or reorganizing and reloading data in units of RDAREAs using `pdrorg`	—	—	—	—	Y	—	—
	Creating plug-in index in delayed batch creation mode	—	—	—	—	Y	—	—
	Releasing used free index storage pages that resulted from addition, deletion, or updating of a large amount of data	—	—	—	—	—	Y*	Y*
	Reinitializing an RDAREA to which only indexes were stored using `pdmod` and then restoring the indexes from the RDAREA in the initialized status	—	—	—	—	—	Y	—
	Creating an index in uncreated status after executing `CREATE INDEX` specifying `EMPTY`	—	—	—	—	—	Y	—
	Re-creating a plug-in index	—	—	—	—	—	Y	—
	Specifying `-i x` during the execution of `pdload`	—	—	—	—	—	Y	—

Legend:

A: Reorganize tables.

B: Unload tables.

C: Reload to tables.

D: Reorganize data dictionary tables.

E: Create indexes in batch.

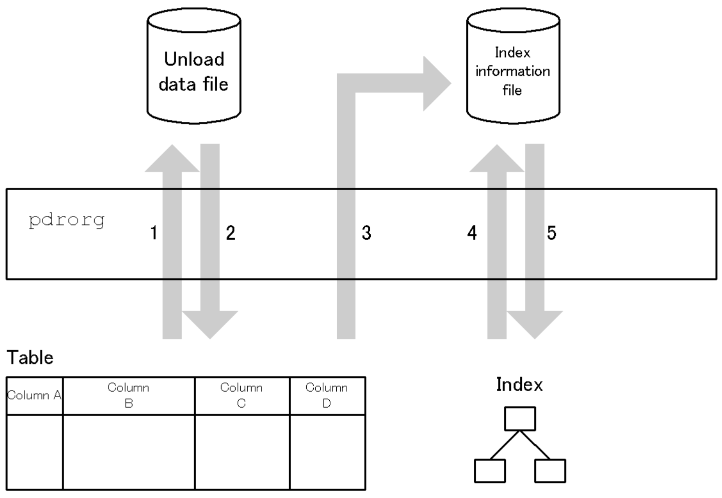
F: Re-create indexes.

G: Reorganize indexes.

Y: `pdrorg`'s function corresponding to the use

* Normally execute index reorganization. Index re-creation involves a data search in the table, but index reorganization does not; therefore, processing speed is faster with index reorganization.

Figure 8-1: Overview of pdrorg functions



Explanation:

Reorganizing table: 1 and 2

Unloading table: 1

Reloading table: 2

Reorganizing data dictionary table: 1 and 2

Creating indexes in batch mode: 5

Re-creating indexes: 3 and 5

Reorganizing indexes: 4 and 5

8.1.2 Reorganizing a table

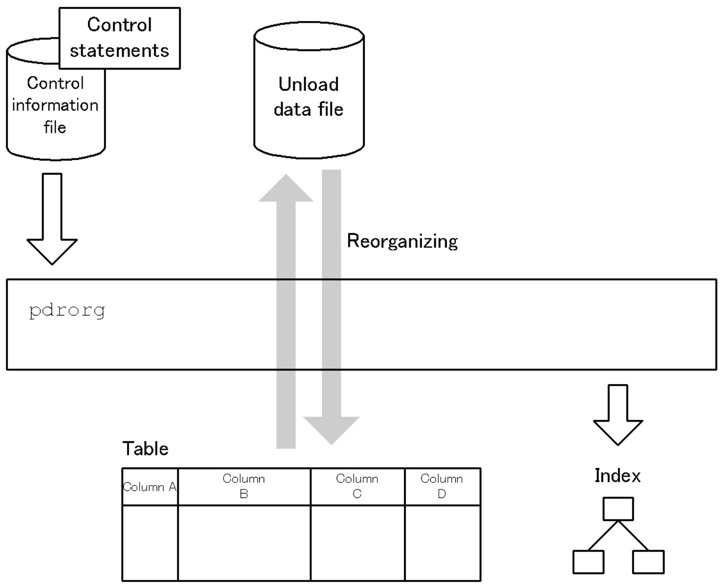
(1) What is reorganizing a table?

Repeated data addition and deletion to a table affects the arrangement of rows in a table, resulting in reduced data access and storage efficiency. In this case, you can improve data access and storage efficiency by saving the table data in a file and then storing it back into the table. This is called reorganizing a table.

You can reorganize a table in units of tables or RDAREAs (for a row-partitioned table).

Figure 8-2 shows an overview of table reorganization.

Figure 8-2: Overview of table reorganization



- Unload data file

  The unload data file is used to temporarily save table data.

- Control information file

  The control information file contains `pdrorg`'s control statements.

  The control statements are used to specify an unload data file, index information, and LOB column information.

(2) Reorganizing a table with a LOB column

If a table contains a LOB column, you can reorganize its LOB column structure base table and LOB column at the same time or separately.

A LOB column structure base table is the part of a table without the LOB column. A LOB column is a column of the BLOB data type.

If you reorganize a LOB column structure base table and the LOB column at the same time, specifying the `-j` option improves performance. Apply the reorganization with the `-j` option omitted when you reorganize either a LOB column structure base table or a LOB column.

If an abstract data type provided by a plug-in is stored in a user LOB RDAREA, you can choose to reorganize the abstract data type together with the LOB column structure base table or you can choose to not reorganize the abstract data type.

Figure 8-3 shows the procedure for reorganizing a table with a LOB column.

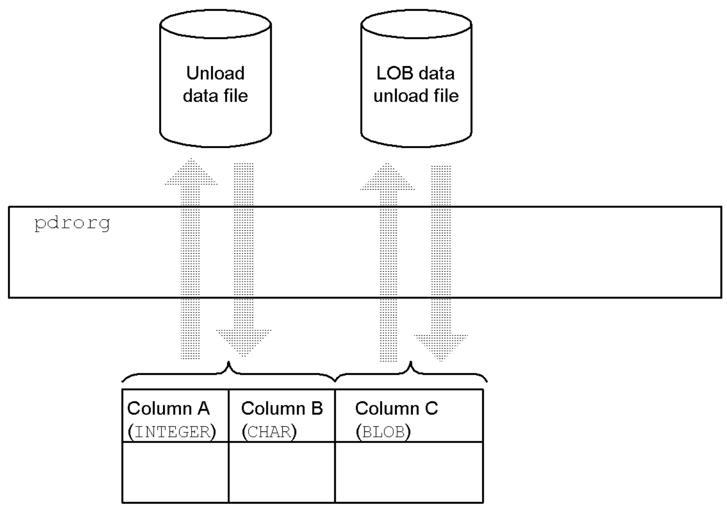


Figure 8-3: Reorganizing a table with a LOB column

Explanation:

Data for the `LOB` column structure base table (Columns A and B) is saved to an unload data file.

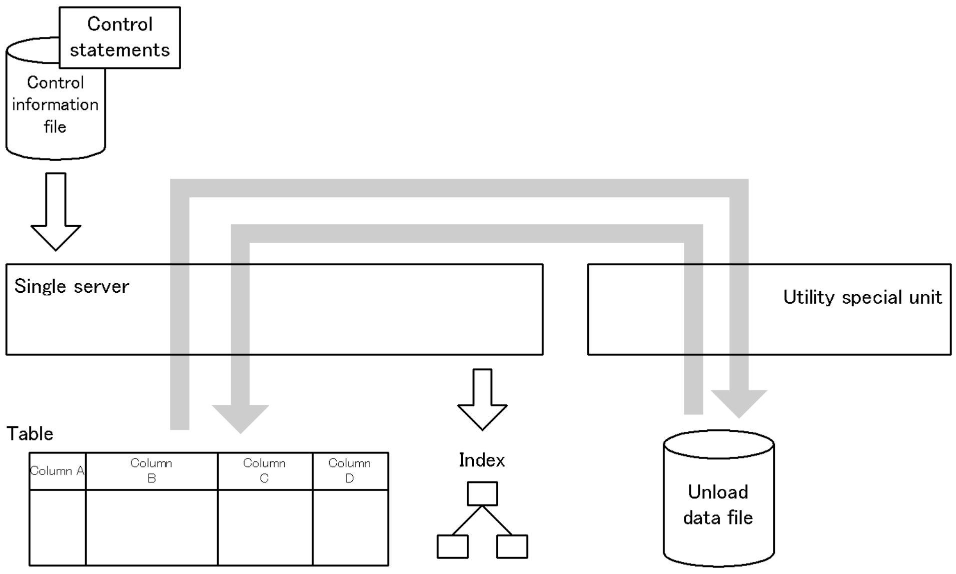
Data for the `LOB` column (Column C) is saved to a `LOB` data unload data file.

By specifying the `-g` and `-j` options, you can save both the `LOB` column structure base table data and the `LOB` column data to an unload data file.

(3) Using a utility special unit

When reorganizing a table with a HiRDB/Single Server, you can use a utility special unit. You can place an unload data file in the utility special unit. Figure 8-4 shows the procedure for reorganizing a table using a utility special unit.

Figure 8-4: Reorganizing a table using a utility special unit



(4) Reorganization with the synchronization point specification

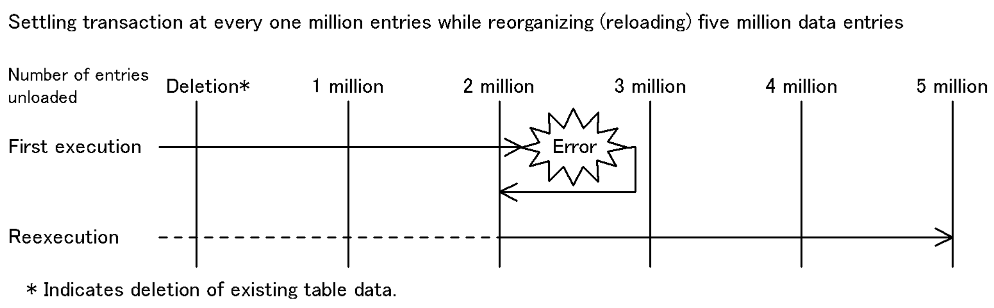
When a table is reorganized, a transaction is normally settled after all data is reloaded. If the utility terminates abnormally during execution, the transaction rolls back to the start point, in which case you have to re-execute the utility from the beginning.

Reorganization with the synchronization point specification enables you to settle the transaction at every specified number of data items. This reduces the time required for rollback and re-execution in the event of abnormal termination.

Note that reorganization with the synchronization point specification is not applicable to unloading, batch index creation, or reorganization of `LOB` columns, in which case a transaction is settled when all processing is completed.

To execute reorganization with the synchronization point specification, specify the `option` statement (with the `job` operand). Figure 8-5 shows an overview of reorganization with the synchronization point specification.

Figure 8-5: Overview of reorganization with the synchronization point specification


Settling transaction at every one million entries while reorganizing (reloading) five million data entries
Number of entries unloaded
Deletion*
1 million
2 million
3 million
4 million
5 million
First execution
Error
Reexecution
* Indicates deletion of existing table data.

Explanation:

At the first execution, rollback occurs after the occurrence of an error. Rollback is to the point of two million entries because the transaction has settled at that point.

The re-execution skips deletion of existing table data and data storage processing up to the point of two million entries, then stores the remaining data.

(5) Reorganization using a UOC

You can use a user-created program (UOC) for reorganization.

When you use a UOC, you can pass data retrieved from the database to the UOC for editing and then output the result, instead of directly saving the retrieved data to the unload data file.

For example, when a large amount of data is to be deleted, suppose that a UOC is used to check the data retrieved from the database to determine whether or not it is to be output to the unload data file. By reloading data from the resulting unload data, you can delete data in the format obtained immediately after reorganization; that is, when there are no scattered free pages (there are no used free pages). Additionally, by using a UOC for unloading, you can output data in a desired format.

8.1.3 Unloading a table

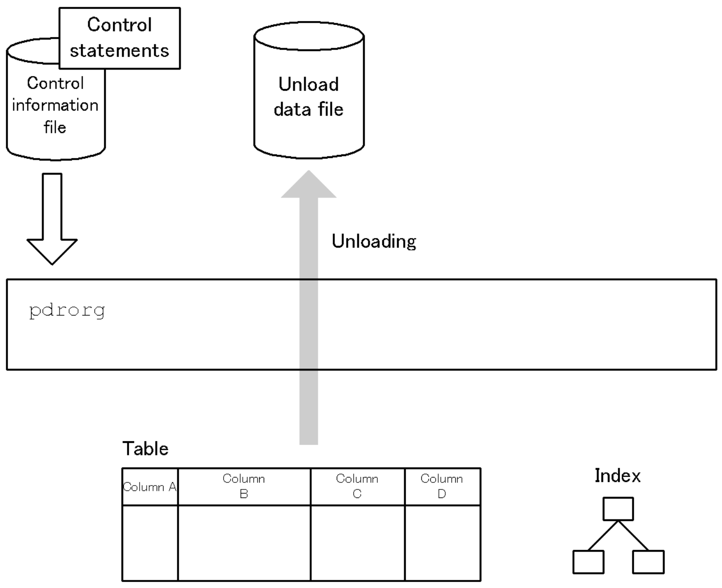
(1) What is unloading a table?

Table reorganization involves saving the table data into a file and then storing the data from the file back into the table. This process for saving table data in a file is called unloading a table.

You can unload a table in units of tables or RDAREAs (applicable to row-partitioned tables).

Figure 8-6 shows an overview of unloading a table.

Figure 8-6: Overview of unloading a table



- Unload data file

  The unload data file is used to save table data.

- Control information file

  The control statement file contains `pdrorg`'s control statements.

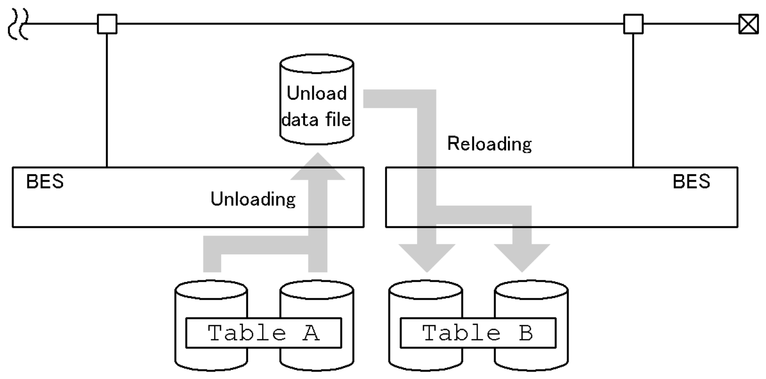
  The control statements are used to specify an unload data file, LOB column information, etc.

(2) Migrating data into another table

To migrate data into another table, first save the table data to an unload data file and then reload the data from the unload data file into the other table. Note that the target table must have the same attribute (`FIX` or non-`FIX`) and column definitions (such as number of columns and their data types) as the source table. However, data migration may still be possible even when their definitions are different if you use `pdload`'s input data file (`-W` option specified) as the unload data.

Figure 8-7 provides an overview of migrating data to another table.

Figure 8-7: Overview of migrating data into another table

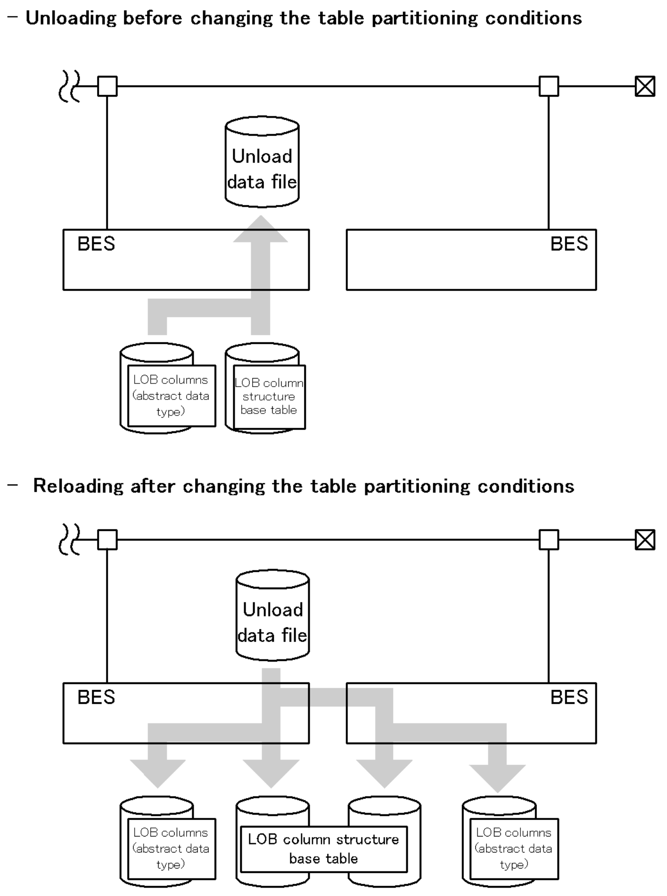


(3) Modifying table partitioning conditions

To modify table partitioning conditions, first save the table data to an unload data file, then modify the table partitioning conditions, and reload the data from the unload data file back into the table.

Figure 8-8 shows an overview of modifying table partitioning conditions.

Figure 8-8: Overview of modifying table partitioning conditions


– Unloading before changing the table partitioning conditions
Unload data file
BES
BES
LOB columns (abstract data type)
LOB column structure base table
– Reloading after changing the table partitioning conditions
Unload data file
BES
BES
LOB columns (abstract data type)
LOB column structure base table
LOB columns (abstract data type)

(4) Using an unload data file as pdload's input data file or with a UAP

To use an unload data file as `pdload`'s input data file or with a UAP, specify the `-W` option during unload operation.

(5) Unloading a table with LOB columns or abstract data type columns provided by a plug-in (LOB attribute)

As with table reorganization, you can unload a LOB column structure base table and LOB columns at the same time or separately.

(6) Unloading a table using a UOC

For details about unloading tables using a UOC, see *8.1.2 (5) Reorganization using a UOC*.

(7) Using a utility special unit

For details about how to use a utility special unit, see Section *8.1.2 Reorganizing a table*.

8.1.4 Reloading to a table

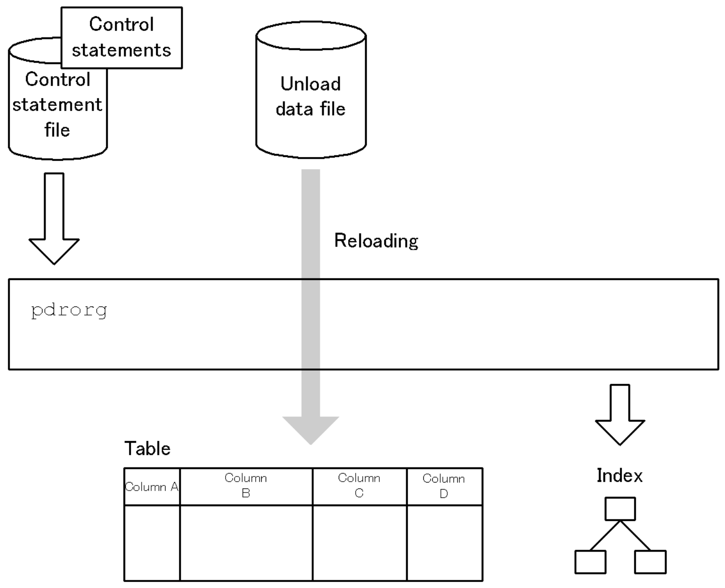
(1) What is reloading to a table?

Table reorganization involves saving table data into a file and then storing the data from the file back into the table. The process for storing data from a file back into a table is called reloading a table.

You can reload a table in units of tables or RDAREAs (applicable to row-partitioned tables).

Figure 8-9 shows an overview of reloading a table.

Figure 8-9: Overview of reloading a table



- Unload data file

  The unload data file is used to save table data.

- Control information file

  The control statement file contains `pdrorg`'s control statements.

  The control statements are used to specify an unload data file, index information, and LOB column information.

(2) Migrating data into another table

For details about how to migrate data into another table, see Section *8.1.3 Unloading a table*.

(3) Modifying table partitioning conditions

For details about how to modify table partitioning conditions, see Section *8.1.3 Unloading a table*.

(4) Reloading to a table with LOB columns or abstract data type columns provided by a plug-in (LOB attribute)

As with table reorganization, you can reload to a LOB column structure base table and LOB columns at the same time or separately.

(5) Using a utility special unit

For details about how to use a utility special unit, see Section *8.1.2 Reorganizing a table*.

(6) Reloading with the synchronization point specification

For details about how to reload with the synchronization point specification, see Section *8.1.2 Reorganizing a table*.

8.1.5 Reorganizing a dictionary table

You reorganize a data dictionary table in the same way you reorganize a user-defined table, by placing the data dictionary RDAREAs (including the data dictionary LOB RDAREA, registry RDAREA, and registry LOB RDAREA, if they are defined) in command shutdown status.

You can reorganize a specific data dictionary table or all data dictionary tables.

If a reload operation results in an error during table reorganization, you can re-execute only the reload operation.

8.1.6 Creating indexes in batch mode

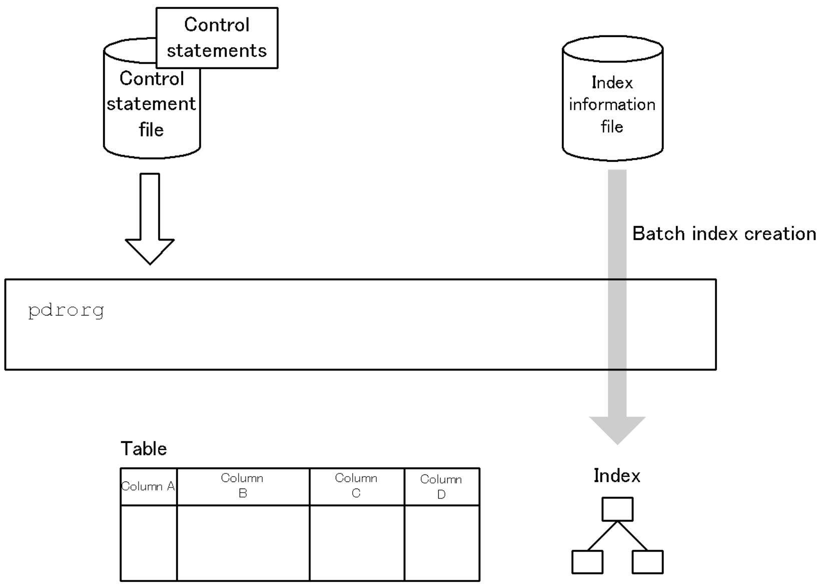
You can create an index using an index information file created with `pdload`, `pdrorg`, or the plug-in index delayed batch creation facility. This is called batch index creation.

If index reorganization results in an error (applicable when `-l a` or `-l p` is specified), you need to execute batch index creation.

You can execute batch index creation in units of indexes or index storage RDAREAs.

Figure 8-10 shows an overview of batch index creation.

Figure 8-10: Batch index creation



- Index information file

  The index information file contains information about an index that was output by `pdload` or `pdrorg`. If the plug-in index delayed batch creation facility is used, this file contains the index information updated by a UAP.

- Control information file

  The control information file contains `pdrorg`'s control statements.

  The control statements are used to specify index information.

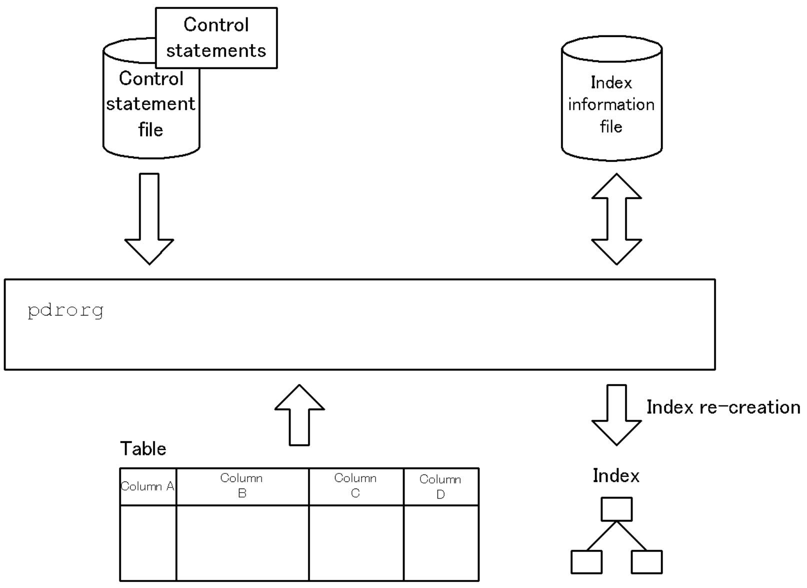
8.1.7 Re-creating an index

You can create an index by searching table data and generating index information. This is called index re-creation.

You can re-create an index in units of indexes or index storage RDAREAs.

Figure 8-11 shows an overview of index re-creation.

Figure 8-11: Overview of index re-creation



- Index information file

  The index information file contains information about an index that was generated by searching table data.

- Control statement file

  The control statement file contains `pdrorg`'s control statements.

  The control statements are used to specify index information.

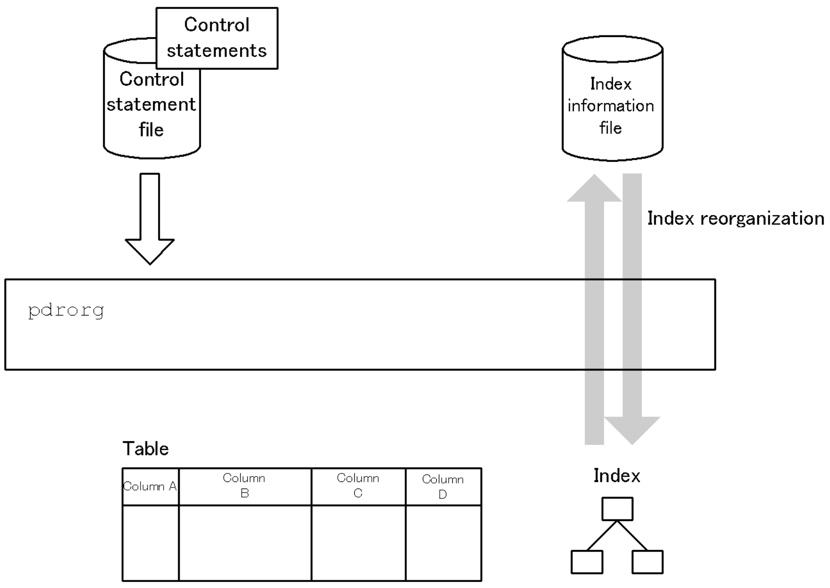
8.1.8 Reorganizing an index

You can create an index information file by searching index key information and rearranging the index on the basis of that information. This is called index reorganization.

You can reorganize an index in units of indexes or index storage RDAREAs.

Figure 8-12 shows an overview of index reorganization.

Figure 8-12: Overview of index reorganization



- Index information file

  The index information file contains information about an index that was generated by searching index key information.

- Control statement file

  The control statement file contains `pdrorg`'s control statements.

  The control statements are used to specify index information.

8.1.9 Log acquisition mode during execution of pdrorg

When you execute `pdrorg`, we recommend that you use pre-update log acquisition (`-l p`) as the log acquisition mode. The pre-update log acquisition mode is the default.

In the pre-update log acquisition mode, the utility does not acquire a database update log during update processing, thereby reducing the processing time. Compared to when the log acquisition mode is used, you can reduce the utility execution time.

8.1.10 RDAREAs containing a table and indexes being processed

To avoid a UAP accessing a table and an index from unnecessarily being placed in wait status, you should use the `pdhold` command to shut down the RDAREAs that contain the table and index being processed.

8.1.11 Whether or not execution of pdrorg is permitted on a table containing columns of an abstract data type

This section describes the `pdrorg` functions that can be executed on a table containing an abstract data type. Table 8-2 shows whether or not execution of `pdrorg` functions is permitted on a table containing an abstract data type.

Table 8-2: Whether or not execution of pdrorg functions is permitted on a table containing an abstract data type

Function	Condition			Executability
Table reorganization (`-k rorg`, `unld`, and `reld`)	Abstract data type provided by plug-in	No LOB attribute		Y
		With LOB attribute	Only LOB column structure base table	Y
			Concurrently (`-j` option specified)	Y[1]
			Only LOB columns	NE
	User-defined abstract data type			NE
Index creation (`-k ixmk` and `ixrc`)	Plug-in index created from the index type provided by plug-in			Y
	Index of user-defined abstract data type			NE
Index reorganization (`-k ixor`)	—			NE
Reloading to another table (`tblname` statement) Modifying table partitioning conditions	Abstract data type provided by plug-in			Y[2]
	User-defined abstract data type			NE
Creation of input data file for `pdload` (`-W` option)	Abstract data type provided by plug-in			Y[3]
	User-defined abstract data type			NE

Y: Executable.

NE: Not executable.

—: No condition.

[1] Executable only for an abstract data type provided by a plug-in with the unload facility.

[2] With some plug-ins, the function is not executable unless you specify the constructor

parameter reverse creation function. For details about the function name to be specified, see the applicable plug-in documentation.

[3] With some plug-ins, the function is executable only when the constructor parameter reverse creation function is specified. For details about the function name to be specified , see the applicable plug-in documentation.

8.1.12 Executor

The following table describes the privileges required in order to execute each function of `pdrorg`:

pdrorg function	Table subject to processing		
	Data dictionary table	User-defined table	Audit trail table
Table reorganization	`DBA` privilege	`SELECT`, `INSERT`, and `DELETE` privileges or `DBA` privilege	`SELECT`, `INSERT`, and `DELETE` privileges
Table unloading	`DBA` privilege	`SELECT` or `DBA` privilege	`SELECT` privilege
Table reloading	N	`INSERT`, `DELETE`, or `DBA` privilege	`INSERT` and `DELETE` privileges
Reorganization, unloading, and reloading in units of schemas	N	`DBA` privilege or schema's owner	Audit privilege
Batch index creation	N	No privilege required	No privilege required
Index re-creation	N	No privilege required	No privilege required
Index reorganization	N	No privilege required	No privilege required

N: Cannot be executed.

Note

When the Directory Server linkage facility is used, the executor must have as its role the indicated privilege as a `pdrorg` executor.

8.2 Reorganizing a table

8.2.1 Examples

The section presents examples of using the database reorganization utility (reorganizing a table), listed as follows:

Example	Description	Classification
1	Reorganizing a row-partitioned table in units of tables	S
2	Reorganizing a row-partitioned table in units of RDAREAs	S
3	Reorganizing a table with LOB columns • Reorganizing LOB column structure base tables and LOB columns at the same time	S
4	Reorganizing a table with an abstract data type (`SGMLTEXT` type) • Reorganizing only the LOB column structure base table	S
5	Reorganizing a row-partitioned table in units of tables	P
6	Reorganizing a table with LOB columns • Reorganizing LOB column structure base tables and LOB columns at the same time	P
7	Reorganizing a table using EasyMT as an unload data file • Using EasyMT for an unload data file • Creating one unload data file at a single server	P
8	Reorganizing tables in units of schemas	P

S: HiRDB/Single Server

P: HiRDB/Parallel Server

(1) Reorganizing a row-partitioned table in units of tables

Example 1

This example reorganizes a row-partitioned table (`TABLE1`) in units of tables.

The example assumes that the table (`TABLE1`) and indexes are defined as follows:

- Table definition:

```
CREATE TABLE TABLE1(C1 INT NOT NULL,C2 CHAR(8),C3 INT)
       IN ((PDBUSER01) C1 > 10,(PDBUSER02))
```

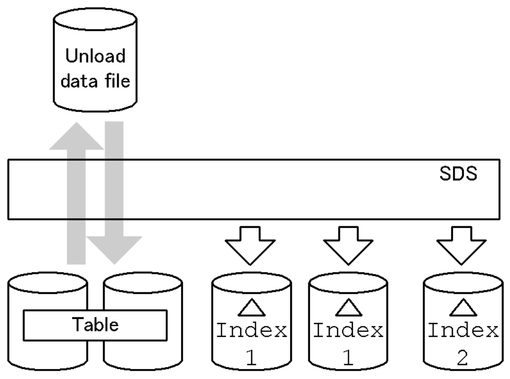
- Index definition (partitioning key index):

```
CREATE INDEX INDEX1 ON TABLE1(C1)
       IN ((PDBUSER03),(PDBUSER04))
```

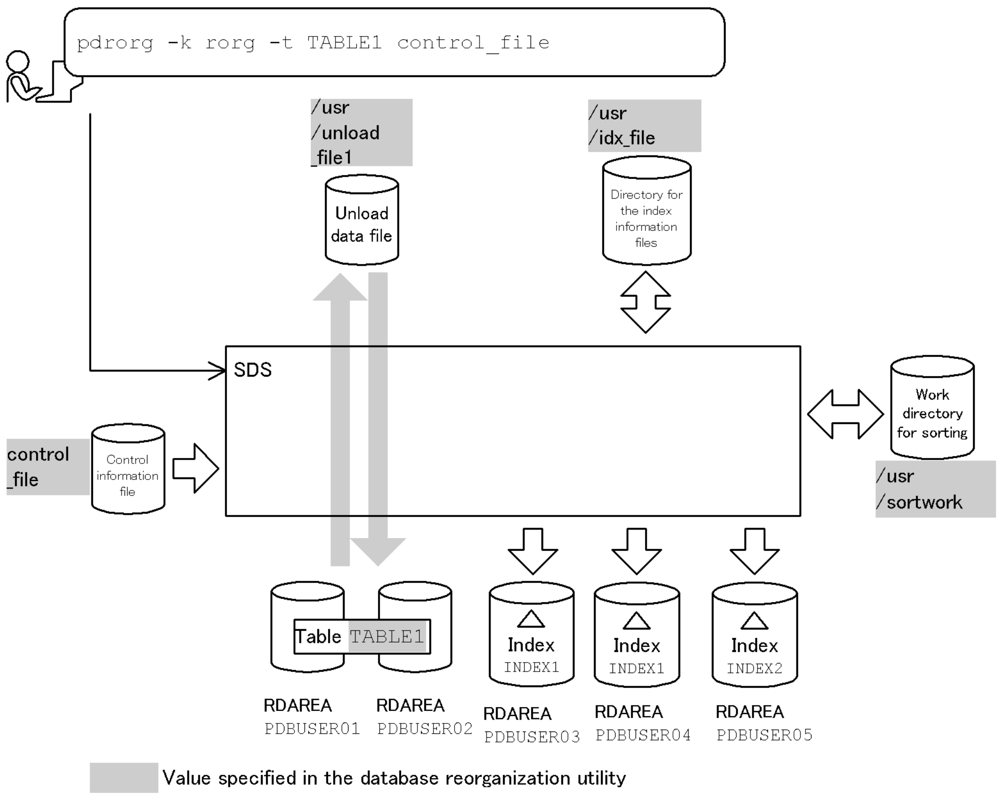
- Index definition (non-partitioning key index):

```
CREATE INDEX INDEX2 ON TABLE1(C2,C1) IN (PDBUSER05)
```

Overview



Relationship between input/output files and RDAREAs



Explanation of the command

This example reorganizes the row-partitioned table (`TABLE1`) in units of tables.

`-k rorg`: Specification for reorganization

`-t TABLE1`: Name of the table being reorganized

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload /usr/unload_file1                    1
idxwork /usr/idx_file                       2
sort /usr/sortwork,8192                     3
```

Explanation:

1. Specifies the unload data file:

`/usr/unload_file1`: Name of the unload data file

2. Specifies the directory in which index information files are to be created:

   `/usr/idx_file`: Name of the directory in which index information files are created

3. Specifies the work directory for sorting:

   `/usr/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)

(2) Reorganizing a row-partitioned table in units of RDAREAs

Example 2

This example reorganizes a row-partitioned table (`TABLE1`) in units of RDAREAs.

The example assumes that the table (`TABLE1`) and indexes are defined as follows:

- Table definition:

```
CREATE TABLE TABLE1(C1 INT NOT NULL,C2 CHAR(8),C3 INT)
       IN ((PDBUSER01) C1 > 10,(PDBUSER02))
```

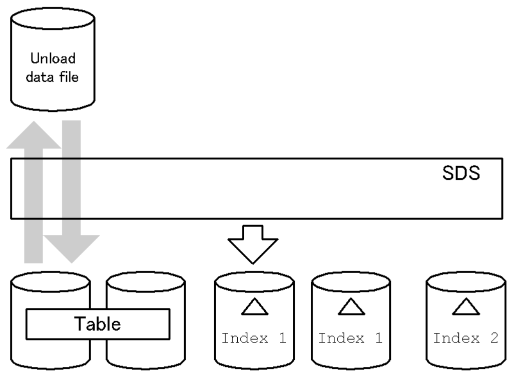
- Index definition (partitioning key index):

```
CREATE INDEX INDEX1 ON TABLE1(C1)
       IN ((PDBUSER03),(PDBUSER04))
```

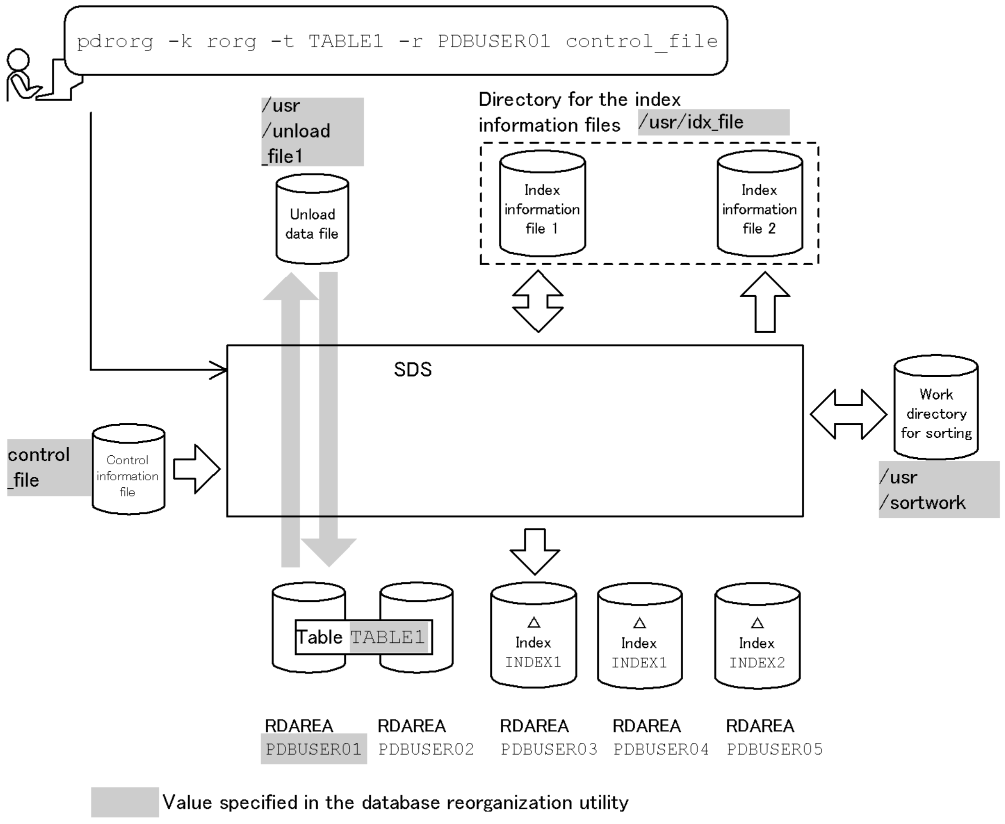
- Index definition (non-partitioning key index):

```
CREATE INDEX INDEX2 ON TABLE1(C2,C1) IN (PDBUSER05)
```

Overview



Relationship between input/output files and RDAREAs



(a) Reorganizing TABLE1 (RDAREA PDBUSER01)

Explanation of the command

This example reorganizes the row-partitioned table (`TABLE1`) in units of RDAREAs.

`-k rorg`: Specification for reorganization

`-t TABLE1`: Name of the table being reorganized

`-r PDBUSER01`: Name of the RDAREA being reorganized

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload /usr/unload_file1                  1
idxwork /usr/idx_file                     2
sort /usr/sortwork,8192                   3
```

Explanation:

1. Specifies the unload data file:

   `/usr/unload_file1`: Name of the unload data file

2. Specifies the directory in which index information files are to be created:

   `/usr/idx_file`: Name of the directory in which index information files are created

3. Specifies the work directory for sorting:

   `/usr/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)

(b) Creating INDEX2

You cannot use `INDEX2` as is. To use `INDEX2`, you need to reorganize RDAREA `PDBUSER02` and output index information for `PDBUSER02`. Create indexes from this index information file and the existing index information file 2 in batch mode by executing `pdrorg` (specifying `-k ixmk`).

For details, see Section *8.6 Creating indexes in batch mode*.

(3) Reorganizing a table with LOB columns

Example 3

For a table with LOB columns (`TABLE2`), this example reorganizes its LOB column structure base table and LOB columns at the same time.

The example assumes that the table and index are defined as follows:

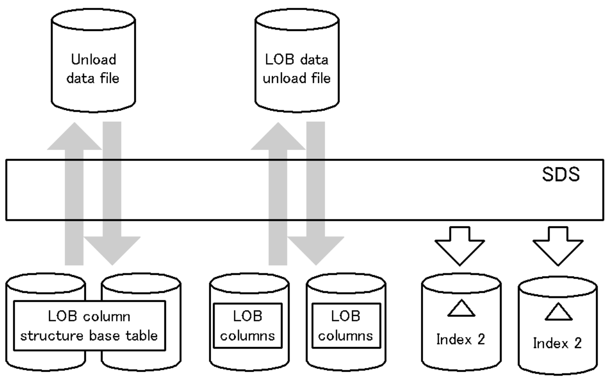
- Table definition:

```
CREATE TABLE TABLE2(C1 INT NOT NULL,C2 BLOB IN
             ((LOBUSER01), (LOBUSER02))) IN
             ((PDBUSER01) C1 > 10,(PDBUSER02))
```

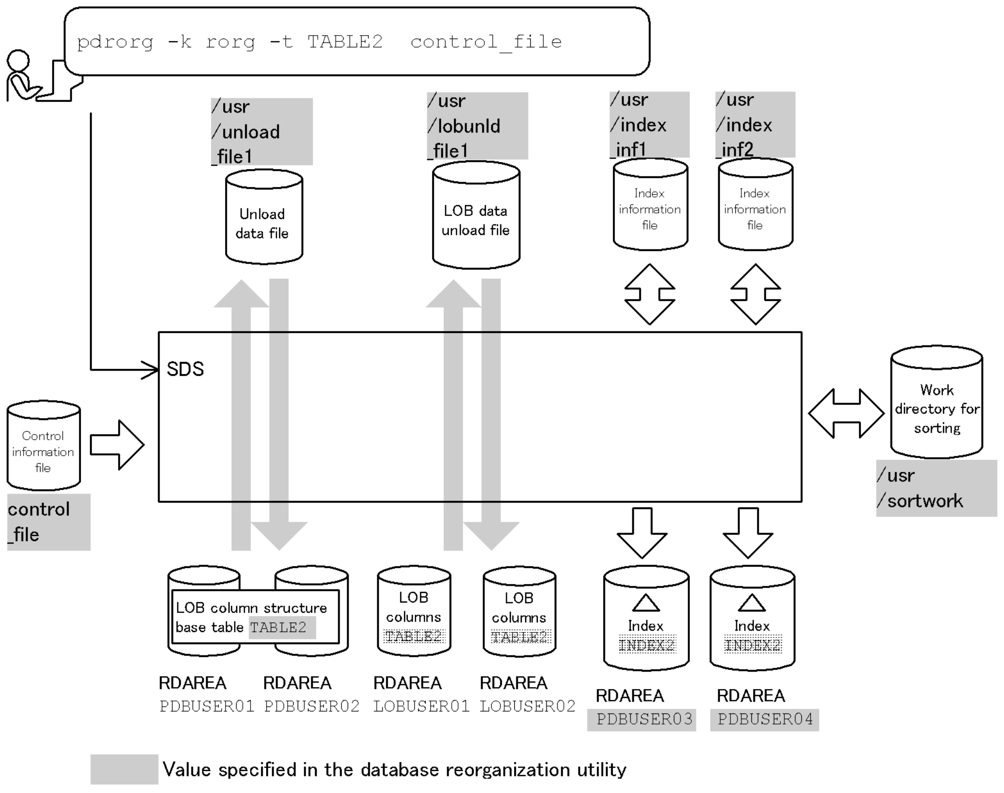
- Index definition:

```
CREATE INDEX INDEX2 ON TABLE2(C1) IN
             ((PDBUSER03),(PDBUSER04))
```

Overview



Relationship between input/output files and RDAREAs



Explanation of the command

This example reorganizes the table with LOB columns (`TABLE2`) in units of tables (both LOB column structure base table and LOB columns at the same time).

`-k rorg`: Specification for reorganization

`-t TABLE2`: Name of the table being reorganized

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload /usr/unload_file1                              1
index INDEX2 PDBUSER03 /usr/index_inf1                2
index INDEX2 PDBUSER04 /usr/index_inf2                2
sort /usr/sortwork,8192                               3
lobunld /usr/lobunld_file1                            4
```

Explanation:

1. Specifies the unload data file:

   `/usr/unload_file1`: Name of the unload data file
2. Specifies the index information files to which index information is output:

   `INDEX2`: Index identifier

   `PDBUSER03`, `PDBUSER04`: Names of the index storage RDAREAs

   `/usr/index_inf1`, `/usr/index_inf2`: Names of the index information files
3. Specifies the work directory for sorting:

   `/usr/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)
4. Specifies the LOB data unload file:

   `/usr/lobunld_file1`: Name of the LOB data unload file

(4) Reorganizing a table with an abstract data type

Example 4

This example reorganizes a table (`TABLE1`) with columns of abstract data type (`SGMLTEXT`). The abstract data type (`SGMLTEXT`) is provided by the HiRDB Text Search Plug-in.

The example assumes that the table and index are defined as follows:

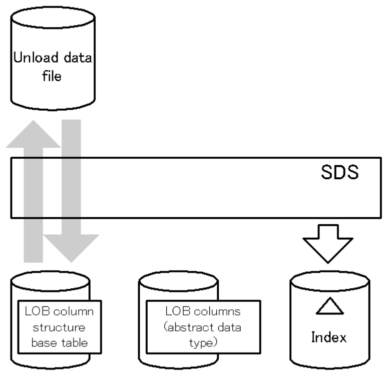
- Table definition:

```
CREATE TABLE TABLE1(C1 INT,C2 SGMLTEXT ALLOCATE
          (SGMLTEXT IN LOBUSER01)
          PLUGIN'<DTD>sgml.dtd</DTD>' ) IN PDBUSER01
```

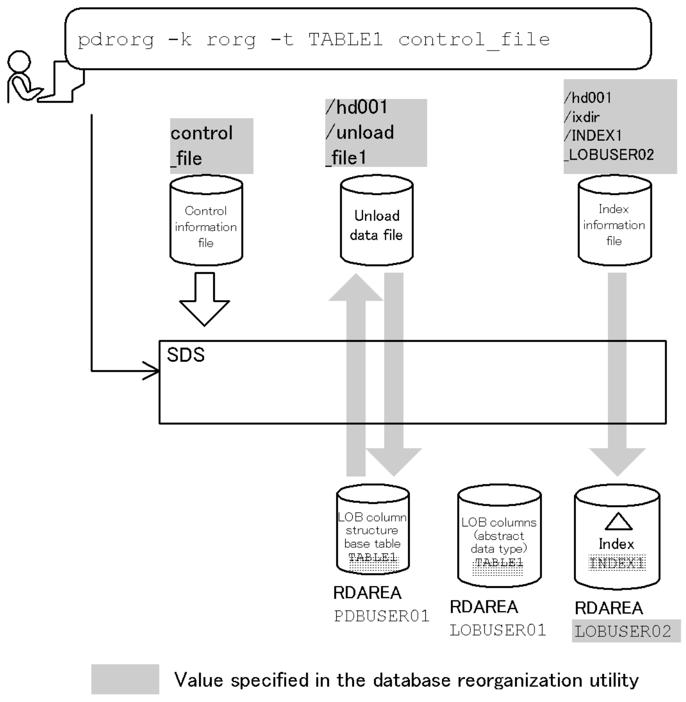
- Plug-in index definition:

```
CREATE INDEX INDEX1 USING TYPE NGRAM ON TABLE1(C2) IN
LOBUSER02
```

Overview


Unload data file
SDS
LOB column structure base table
LOB columns (abstract data type)
Index

Relationship between input/output files and RDAREAs


pdrorg -k rorg -t TABLE1 control_file
control_file
Control information file
/hd001/unload_file1
Unload data file
/hd001/ixdir/INDEX1_LOBUSER02
Index information file
SDS
LOB column structure base table TABLE1
RDAREA PDBUSER01
LOB columns (abstract data type) TABLE1
RDAREA LOBUSER01
Index INDEX1
RDAREA LOBUSER02
Value specified in the database reorganization utility

Explanation of the command

This example reorganizes the table (`TABLE1`).

`-k rorg`: Specification for reorganization

`-t TABLE1`: Name of the table being reorganized

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload /hd001/unload_file1                                    1
index INDEX1 LOBUSER02 /hd001/ixdir/INDEX1_LOBUSER02          2
```

Explanation:

1. Specifies the unload data file:

   `/hd001/unload_file1`: Name of the unload data file

2. Specifies the index information file:

   `INDEX1`: Index identifier of the plug-in index subject to batch index creation

   `LOBUSER02`: Name of the index storage RDAREA for the plug-in index subject to batch index creation

   `/hd001/ixdir/INDEX1_LOBUSER02`: Name of the index information file

(5) Reorganizing a row-partitioned table in units of tables

Example 5

This example reorganizes a row-partitioned table (`TABLE1`) in units of tables.

The example assumes that the table and indexes are defined as follows:

- Table definition:

```
CREATE TABLE TABLE1(C1 INT NOT NULL,C2 CHAR(8),C3 INT)
             IN ((PDBUSER01) C1 > 10,(PDBUSER02))
```

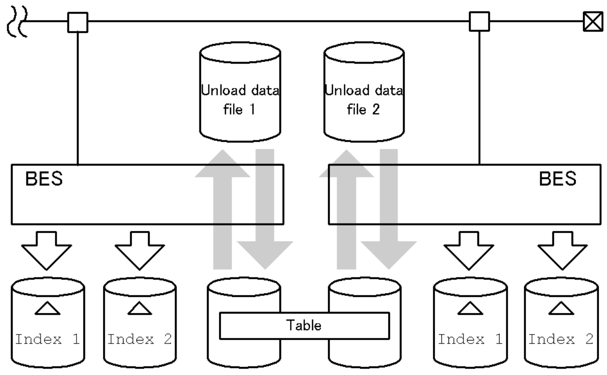
- Index definition (partitioning key index):

```
CREATE INDEX INDEX1 ON TABLE1(C1)
             IN ((PDBUSER03),(PDBUSER05))
```

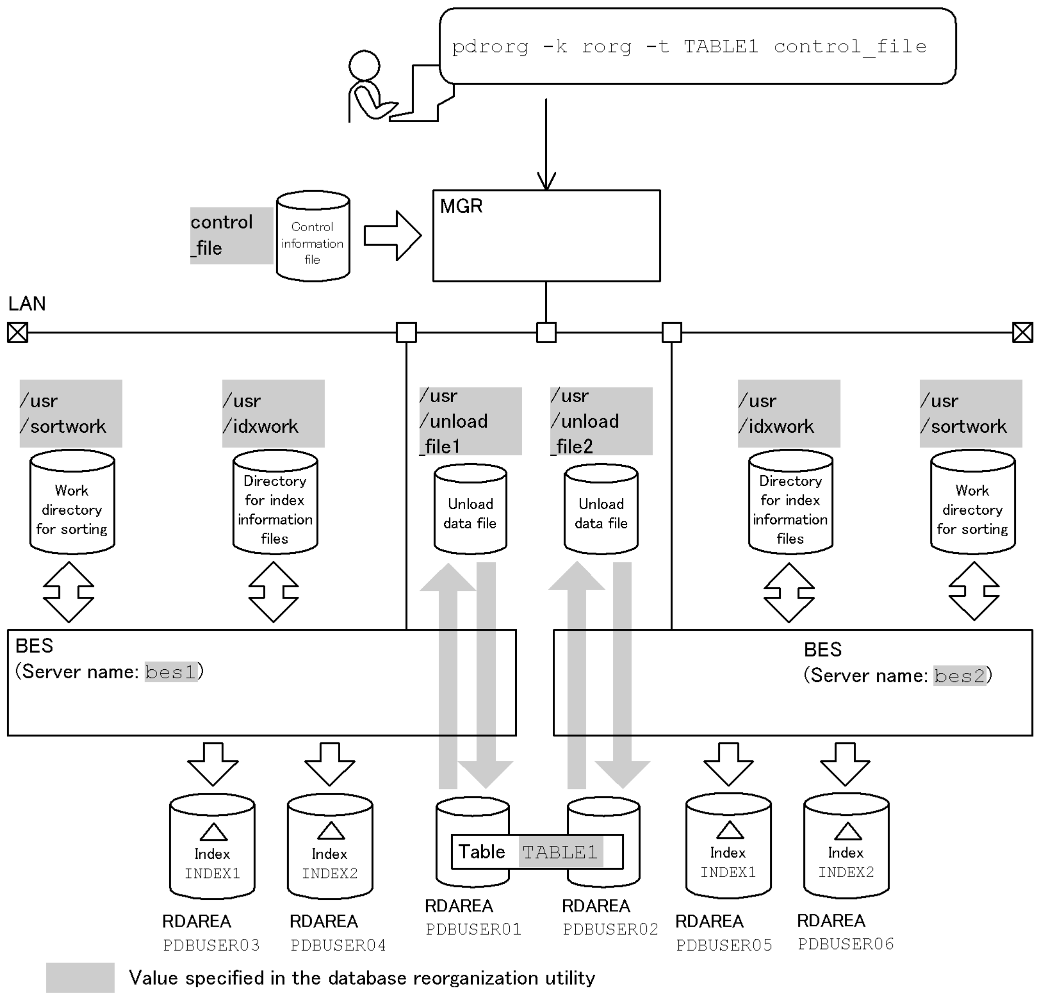
- Index definition (non-partitioning key index):

```
CREATE INDEX INDEX2 ON TABLE1(C2,C1)
             IN ((PDBUSER04),(PDBUSER06))
```

Overview



Relationship between input/output files and RDAREAs



Explanation of the command

This example reorganizes the row-partitioned table (`TABLE1`) in units of tables.

`-k rorg`: Specification for reorganization

`-t TABLE1`: Name of the table being reorganized

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/usr/unload_file1                 1
idxwork bes1 /usr/idxwork                     2
sort bes1 /usr/sortwork,8192                  3
unload bes2:/usr/unload_file2                 1
idxwork bes2 /usr/idxwork                     2
sort bes2 /usr/sortwork,8192                  3
```

Explanation:

1. Specifies the unload data file:

   `bes1`, `bes2`: Names of the servers containing the unload data files

   `/usr/unload_file1`, `/usr/unload_file2`: Names of the unload data files

2. Specifies the directory for index information files to which index information is to be output:

   `bes1`, `bes2`: Names of the servers used to create index information files

   `/usr/idxwork`: Name of the directory for index information files

3. Specifies the work directory for sorting:

   `bes1`, `bes2`: Names of the servers used to create the sort work file

   `/usr/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)

(6) Reorganizing a table with LOB columns

Example 6

For a table with LOB columns (`TABLE2`), this example reorganizes both the LOB column structure base table and LOB columns at the same time.

The example assumes that the table and index are defined as follows:

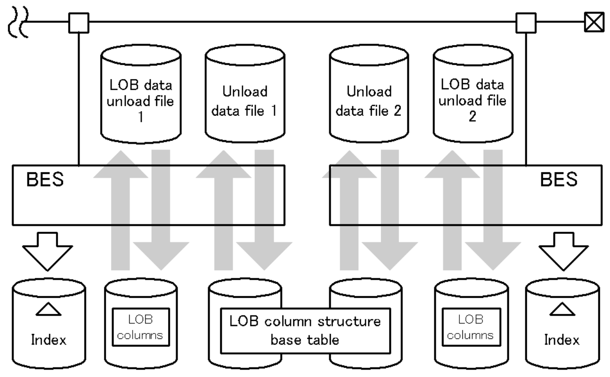
- Table definition:

```
CREATE TABLE TABLE2(C1 INT NOT NULL,C2 BLOB IN
              ((LOBUSER01), (LOBUSER02))) IN
              ((PDBUSER01) C1 > 10,(PDBUSER02))
```

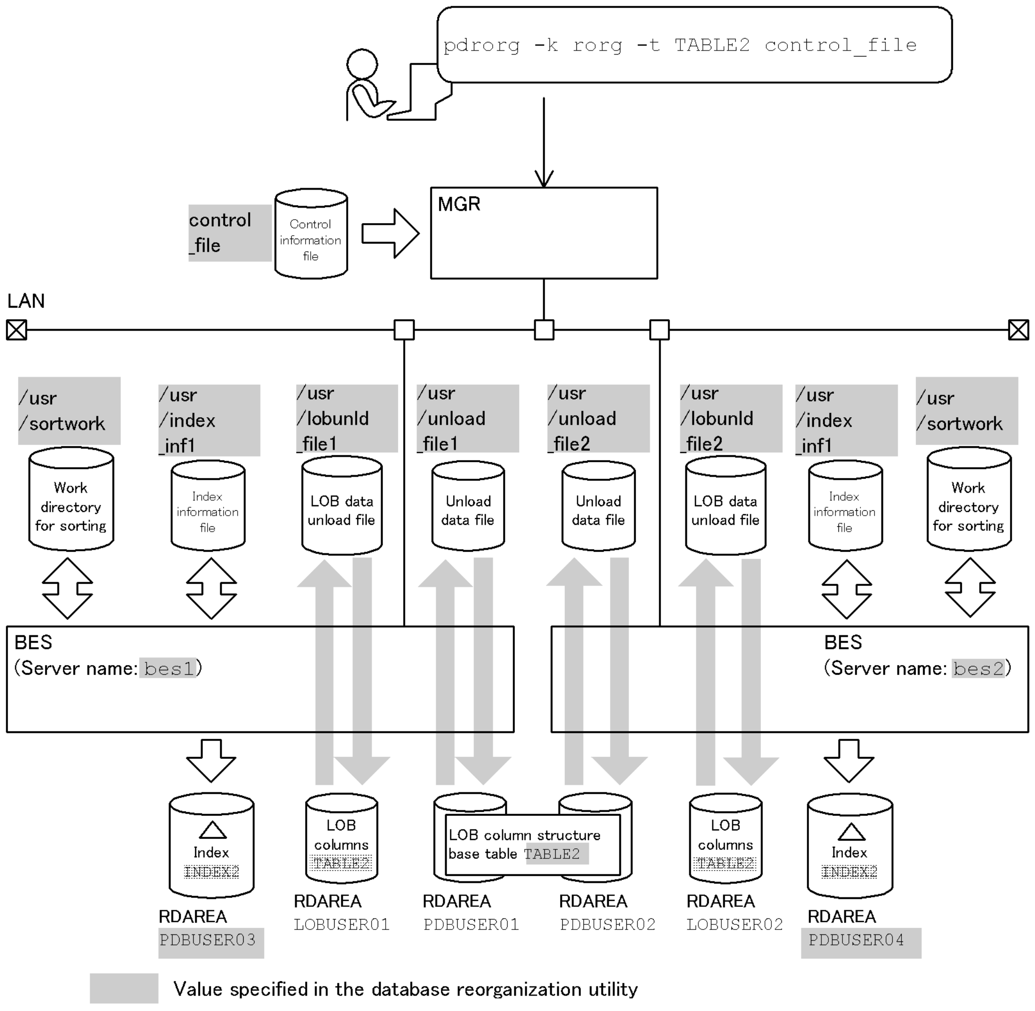
- Index definition:

```
CREATE INDEX INDEX2 ON TABLE2(C1) IN
               ((PDBUSER03),(PDBUSER04))
```

Overview


LOB data unload file 1
Unload data file 1
Unload data file 2
LOB data unload file 2
BES
BES
Index
LOB columns
LOB column structure base table
LOB columns
Index

Relationship between input/output files and RDAREAs



Explanation of the command

This example reorganizes the table with LOB columns (`TABLE2`) in units of tables (both the LOB column structure base table and LOB columns at the same time).

`-k rorg`: Specification for reorganization

`-t TABLE2`: Name of the table being reorganized

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/usr/unload_file1                            1
unload bes2:/usr/unload_file2                            1
index INDEX2 PDBUSER03 /usr/index_inf1                   2
index INDEX2 PDBUSER04 /usr/index_inf2                   2
sort bes1 /usr/sortwork,8192                             3
sort bes2 /usr/sortwork,8192                             3
lobunld bes1:/usr/lobunld_file1                          4
lobunld bes2:/usr/lobunld_file2                          4
```

Explanation:

1. Specifies the unload data file:

   `bes1`, `bes2`: Names of the servers containing the unload data files

   `/usr/unload_file1`, `/usr/unload_file2`: Names of the unload data files

2. Specifies the index information files to which index information is output:

   `INDEX2`: Index identifier

   `PDBUSER03`, `PDBUSER04`: Names of the index storage RDAREAs

   `/usr/index_inf1`, `/usr/index_inf2`: Names of the index information files

3. Specifies the work directory for sorting:

   `bes1`, `bes2`: Names of the servers used to create the sort work file

   `/usr/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)

4. Specifies the LOB data unload files:

   `bes1`, `bes2`: Names of the servers containing the LOB data unload files

   `/usr/lobunld_file1`, `/usr/lobunld_file2`: Names of the LOB data unload files

(7) Reorganizing a table using EasyMT as an unload data file

Example 7

This example reorganizes a row-partitioned table (TABLE1) in units of tables using EasyMT as an unload data file. It creates only one unload data file specifying the `-g` option.

The example assumes that the table and indexes are defined as follows:

- Table definition:

```
CREATE TABLE TABLE1(C1 INT NOT NULL,C2 CHAR(8),C3 INT)
             IN ((PDBUSER01) C1 > 10,(PDBUSER02))
```

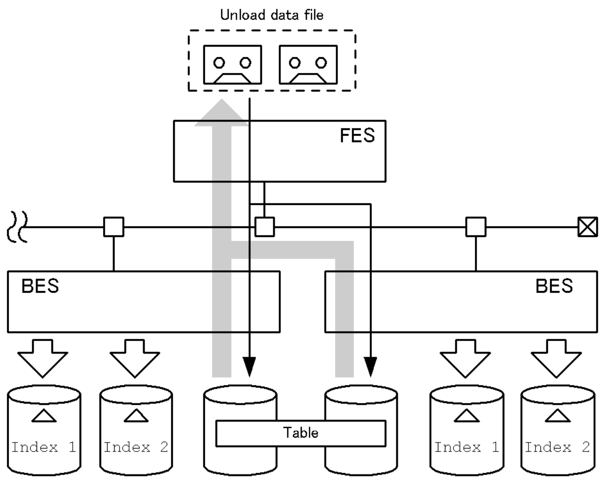
- Index definition (partitioning key index):

```
CREATE INDEX INDEX1 ON TABLE1(C1)
             IN ((PDBUSER03),(PDBUSER05))
```

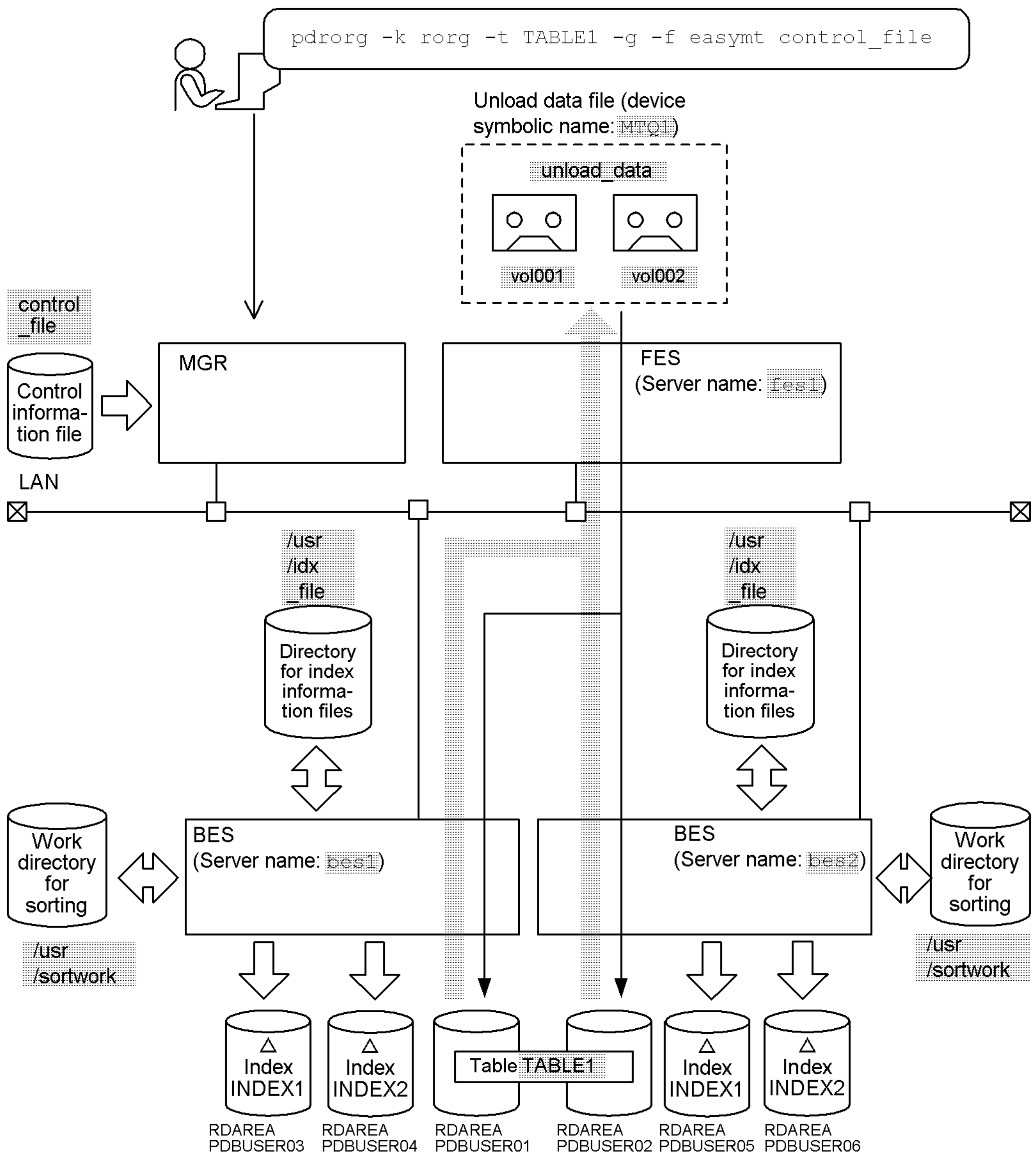
- Index definition (non-partitioning key index):

```
CREATE INDEX INDEX2 ON TABLE1(C2,C1)
             IN ((PDBUSER04),(PDBUSER06))
```

Overview



Relationship between input/output files and RDAREAs



▨ : Value specified in the database reorganization utility
MGR: System manager
FES: Front-end server
BES: Back-end server

Explanation of the command

This example reorganizes the row-partitioned table (TABLE1) in units of tables.

`-k rorg`: Specification for reorganization

`-t TABLE1`: Name of the table being reorganized

`-g`: Specification for integrating data into a single unload data file

`-f easymt`: Specification for using EasyMT as unload data file

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
mtguide use                                                 1
unload fes1:MT01 file=unload_data,vol=(vol001,vol002)       2
idxwork bes1 /usr/idx_file                                  3
idxwork bes2 /usr/idx_file                                  3
sort bes1 /usr/sortwork,8192                                4
sort bes2 /usr/sortwork,8192                                4
```

Explanation:

1. Specifies EasyMT:

   `use`: Specification for using MTguide

2. Specifies the unload data file:

   `fes1`: Name of the server used to create unload data file

   `MT01`: Device symbolic name managed by MTguide

   `unload_data`: Name of the MT file

   `(vol001,vol002)`: Specification for unloading data to two magnetic tape volumes

3. Specifies the directory for index information files to which index information is to be output:

   `bes1, bes2`: Names of the servers used to create index information files

   `/usr/idx_file`: Name of the directory in which index information files are created

4. Specifies the work directory for sorting:

   `bes1, bes2`: Names of the servers used to create sort work file

   `/usr/sortwork`: Name of the directory in which sort work file is created

   `8192`: Size of the buffer for sorting (in KB)

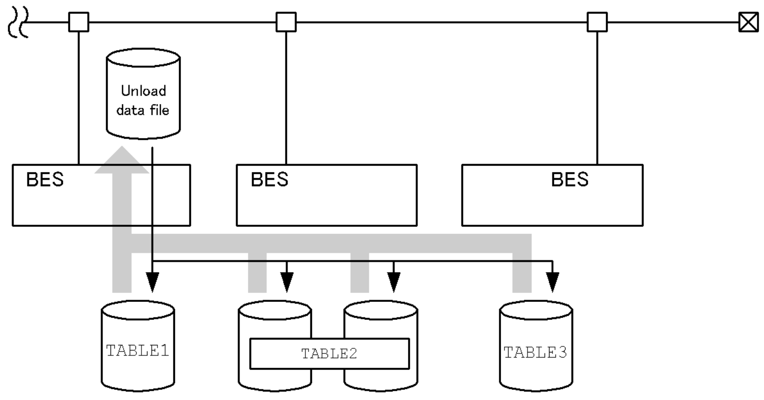
(8) Reorganizing tables in units of schemas

Example 8

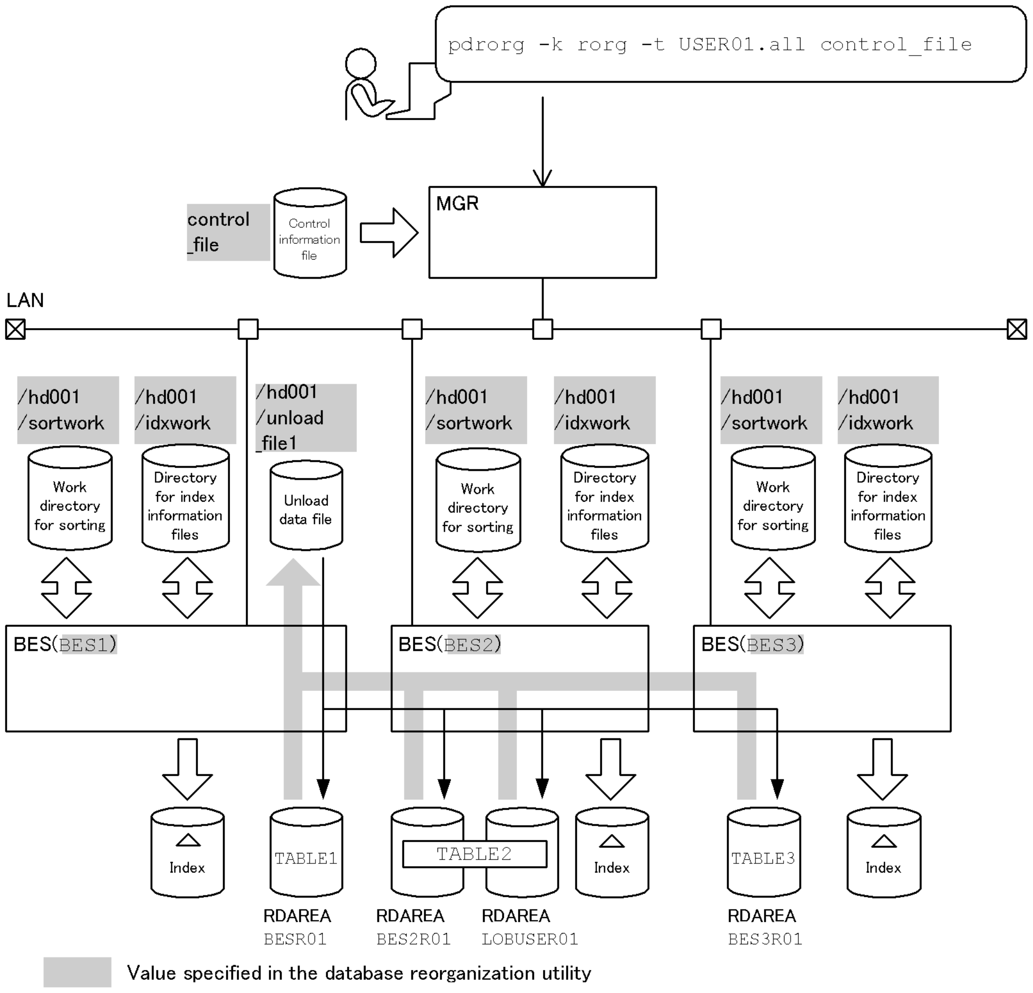
This example reorganizes all tables owned by schema `USER01`, for which the following tables have been defined:

```
CREATE TABLE TABLE1(C1 INT) IN (BES1R01)
CREATE TABLE TABLE2(C1 BLOB IN LOBUSER01) IN(BES2R01)
CREATE TABLE TABLE3(C1 ADT1) IN(BES3R01)
```

Overview



Relationship between input/output files and RDAREAs



Explanation of the command

This example reorganizes all tables owned by schema `USER01`. When reorganizing tables in units of schemas with a HiRDB/Parallel Server, the system assumes the `-g` and `-j` options.

`-k rorg`: Specification for reorganization

`-t USER01.all`: Specification for all the tables owned by schema `USER01`

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/hd001/unload_file1              1
idxwork bes1 /hd001/idxwork                  2
sort bes1 /hd001/sortwork,8192               3
idxwork bes2 /hd001/idxwork                  2
sort bes2 /hd001/sortwork,8192               3
idxwork bes3 /hd001/idxwork                  2
sort bes3 /hd001/sortwork,8192               3
```

Explanation:

1. Specifies the unload data file:

   `bes1`: Name of the server containing the unload data file

   `/hd001/unload_file1`: Name of the unload data file

2. Specifies the directory for index information files to which index information is to be output:

   `bes1`, `bes2`, `bes3`: Names of the servers used to create index information files

   `/hd001/idxwork`: Name of the directory for index information files

3. Specifies the work directory for sorting:

   `bes1`, `bes2`, `bes3`: Names of the servers used to create the sort work file

   `/hd001/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)

8.2.2 Cross-reference by purpose

Required options and control statements depend on the type of reorganization.

The options listed in (1) and the control statements are mandatory. The items in (2) through (4) are related to the options and control statements presented in the *Reference* column. For details about the options, see Section 8.9.2.

(1) Required items

Item	Reference			
	Option		Control statement	
Type of `pdrorg` processing	`-k rorg`	(1)	—	—
Reorganization of a user-defined table (can be omitted because the default is `-c user`)	`-c user`	(2)	—	—

Item	Reference			
	Option		Control statement	
Name of the table to be reorganized	`-t`	(3)	—	—
File containing the control statements	*control-information-filename*	(19)	—	—
Information about an unload data file	—	—	`unload` statement	8.9.4

— : Not applicable

(2) Information to be specified depending on the attribute of the table being reorganized

Item	Reference location			
	Option		Control statement	
Indexes are defined	`-i`	(6)	index statement	8.9.5
	`-o`	(15)	idxwork statement	8.9.7
			sort statement	8.9.8
There are LOB columns	`-j`	(9)	lobunld statement	8.9.9
There are columns of abstract data type provided by a plug-in (LOB attribute)	`-j`	(9)	unld_func statement	8.9.13
			reld_func statement	8.9.14
Consecutive trailing spaces are to be removed from a table with suppress option specified*	`-S`	(14)	—	—
There are constraint definitions	—	—	`constraint` statement	8.9.15
Inner replica facility is used	`-q`	(18)	—	—

— : Not applicable

* This option helps you reduce the processing time and the size of the unload data file.

(3) Information to be specified depending on the type of unload data file or LOB data unload file

Item	Reference location			
	Option		Control statement	
Integrating data into a single file	`-g`	(8)	—	—

Item	Reference location			
	Option		Control statement	
Using EasyMT	`-f`	(12)	mtguide statement	8.9.3
			emtdef statement	8.9.3
Using HiRDB files	`-f`	(12)	—	—
Specifying the unload data storage sequence (in order of cluster key, index, or data storage)	`-b`	(13)	—	—

— : Not applicable

(4) Information to be specified depending on the reorganization method

Item	Reference location			
	Option		Control statement	
Reorganizing only some of the RDAREAs for a row-partitioned table	`-r`	(4)	—	—
Acquiring or not acquiring database update log	`-l`	(5)	—	—
Specifying synchronization points when reorganizing a large amount of data	—	—	option statement	8.9.16
Monitoring the execution time of `pdrorg`	—	—	option statement	8.9.16
Forcibly restoring the table status to normal	—	—	option statement	8.9.16
Changing the `pdrorg` executor's user authorization identifier to a value other than the value of environment variable `PDUSER`[1]	`-u`	(10)	—	—
Reorganizing in batch input/output mode using local buffer instead of global buffer[2]	`-n`	(11)	—	—
Changing a reorganization status message output interval to a value other than 100,000 lines	`-m`	(16)	—	—
Monitoring the response time for server-to-server communication	-X	(17)	—	—
Using the same space character in the entire table data	—	—	option statement	8.9.16
Changing the percentage of free space in table and index during reorganization[3]	—	—	option statement	8.9.16

Item	Reference location			
	Option		Control statement	
Obtaining tuning information during the execution of `pdrorg`	—	—	report statement	8.9.17
Deleting unneeded data during reorganization	—	—	unlduoc statement	8.9.10

— : Not applicable

[1] If omitted, the system assumes the value of the `PDUSER` environment variable. If the `PDUSER` environment variable has not been specified, the system assumes the user name in the login window.

[2] By specifying the number of batch input/output pages, you can reduce the number of I/O operations, because data is input/output in units of specified pages in batch mode.

[3] If a space shortage occurs in an RDAREA during execution of `pdrorg`, you can complete the processing without having to expand the RDAREA temporarily (except for a table or index whose percentage of free space is 0).

8.3 Unloading a table

8.3.1 Examples

The section presents examples of using the database reorganization utility (unloading a table), listed as follows:

Example	Description	Classification
1	Unloading a row-partitioned table in units of tables • Using the unload data file as an input data file for `pdload` • Integrating all data in a single unload data file	P
2	Unloading and reloading a table • After unloading the table, modifying the table partitioning conditions before reloading	
3	Unloading and reloading a table • Using unloaded table data to reload another table	
4	Unloading and reloading a table with an abstract data type (`SGMLTEXT` type) • After unloading the table, modifying the table partitioning conditions before reloading	
5	Unloading and reloading a table with an abstract data type (`SGMLTEXT` type) • Using unloaded table data to reload another table	

P: HiRDB/Parallel Server

(1) Unloading a row-partitioned table in units of tables

Example 1

This example unloads a row-partitioned table (`TABLE1`) in units of tables. It uses the unload data as an input data file (binary format) for `pdload`. All data is integrated in a single data file.

The example assumes that the following table and indexes have been defined:

- Table definition:

```
CREATE TABLE TABLE1(C1 INT NOT NULL,C2 CHAR(8),C3 INT)
             IN ((PDBUSER01) C1 > 10,(PDBUSER02))
```

- Index definition (partitioning key index):

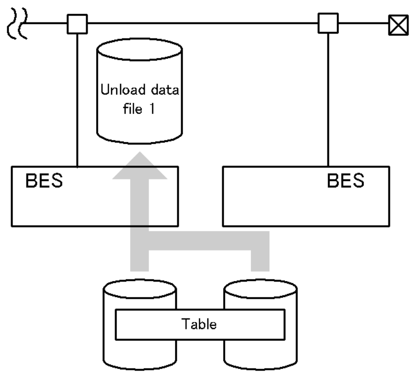
```
CREATE INDEX INDEX1 ON TABLE1(C1)
             IN ((PDBUSER03),(PDBUSER05))
```

- Index definition (non-partitioning key index):

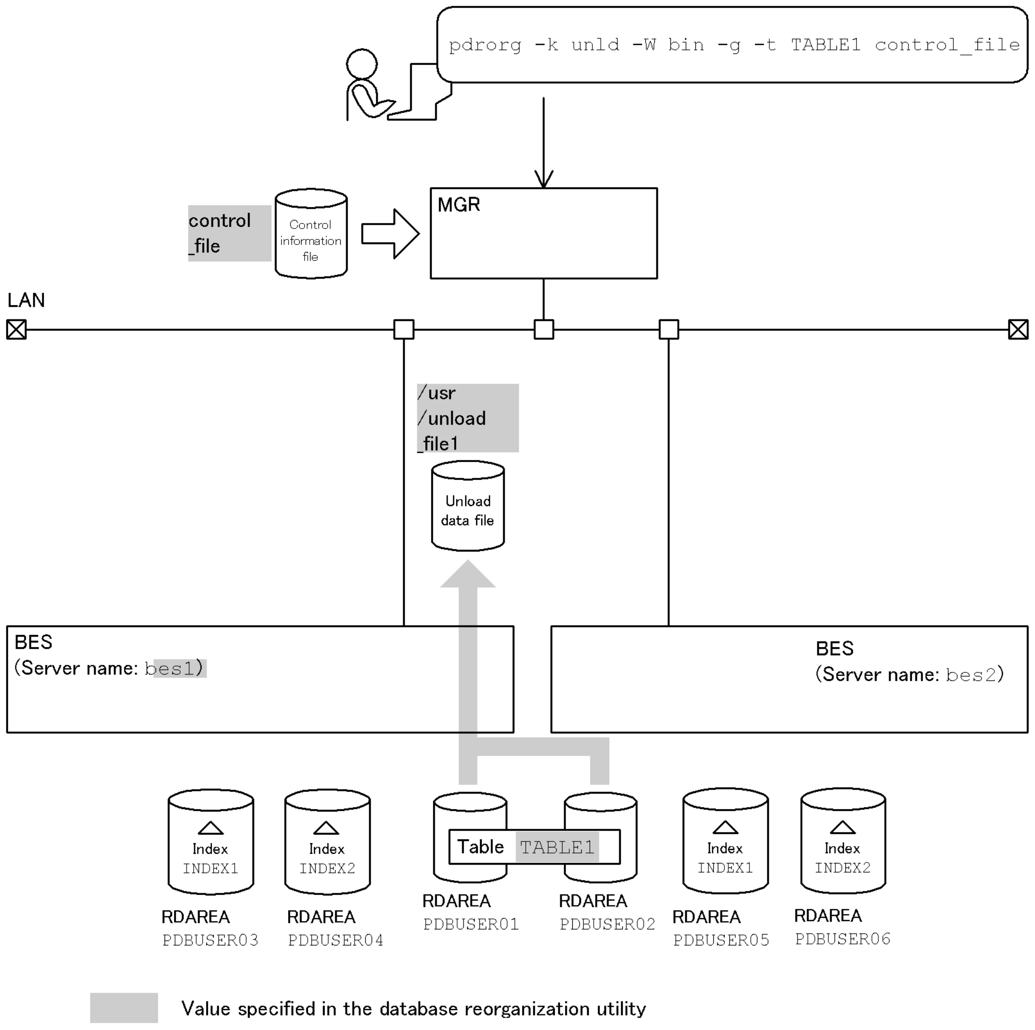
```
CREATE INDEX INDEX2 ON TABLE1(C2,C1)
```

```
                          IN ((PDBUSER04),(PDBUSER06))
```

Overview



Relationship between input/output files and RDAREAs



Explanation of the command

This example unloads the row-partitioned table (`TABLE1`) in units of tables.

`-k unld`: Specification for unloading

`-W bin`: Specification for using the unload data file as an input data file (binary format) for `pdload`

`-g`: Specification for integrating all data in a single unload data file

`-t TABLE1`: Name of the table being unloaded

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/usr/unload_file1                     1
```

Explanation:

1. Specifies the unload data file:

   `bes1`: Name of the server containing the unload data file

   `/usr/unload_file1`: Name of the unload data file

(2) Unloading and reloading a table while modifying the table partitioning conditions

Example 2

To modify a table's (`TABLE1`'s) partitioning conditions, this example first unloads the table, modifies the table partitioning conditions, then reloads the table.

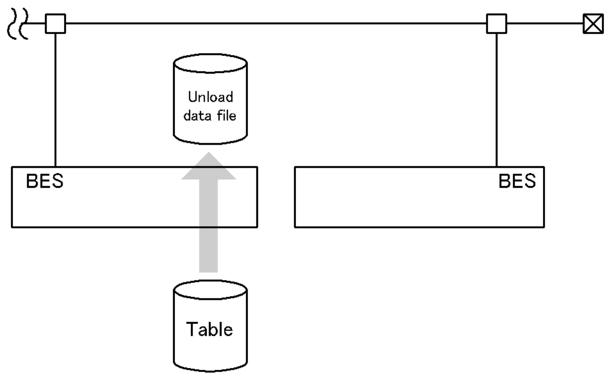
The example assumes that the following table and index have been defined:

- Table definition:
```
CREATE TABLE TABLE1(C1 INT NOT NULL,C2 CHAR(8),C3 INT) IN
PDBUSER01
```

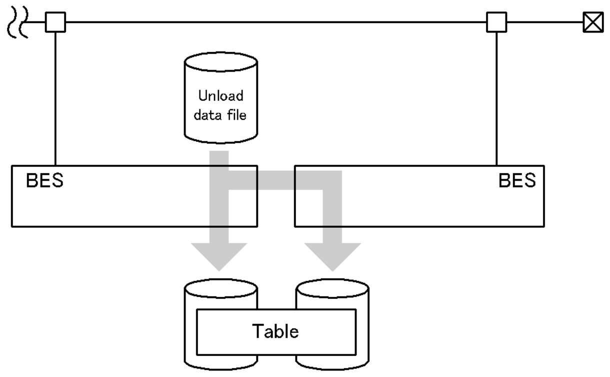
- Index definition (partitioning key index):
```
CREATE INDEX INDEX1 ON TABLE1(C1) IN PDBUSER03
```

Overview

– **Unloading before modifying the table partitioning conditions**

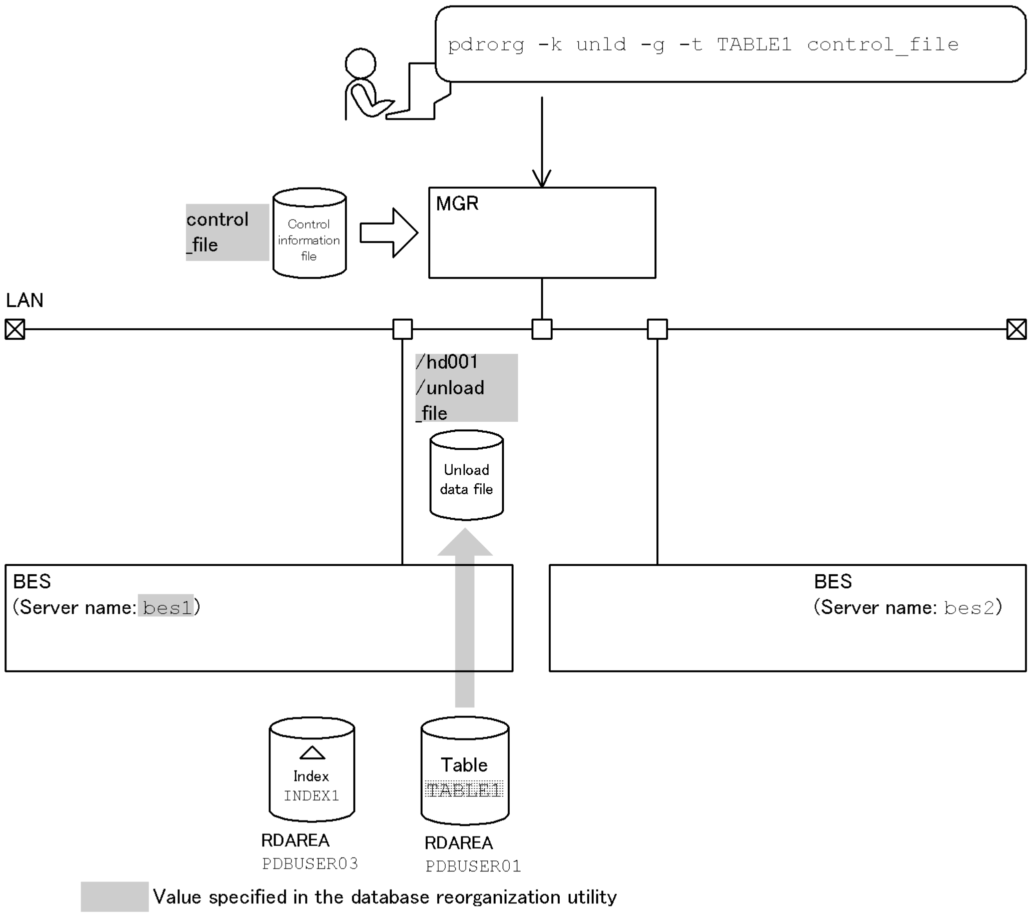


– **Reloading after modifying the table partitioning conditions**



(a) Unloading the table before modifying the table partitioning conditions

Relationship between input/output files and RDAREAs



Explanation of the command

The example unloads the table (`TABLE1`).

`-k unld`: Specification for unloading

`-t TABLE1`: Name of the table being unloaded

`-g`: Specification for modifying the table partitioning conditions

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/hd001/unload_file                          1
```

Explanation:

1. Specifies the unload data file:

   `bes1`: Name of the server containing the unload data file

   `/hd001/unload_file`: Name of the unload data file

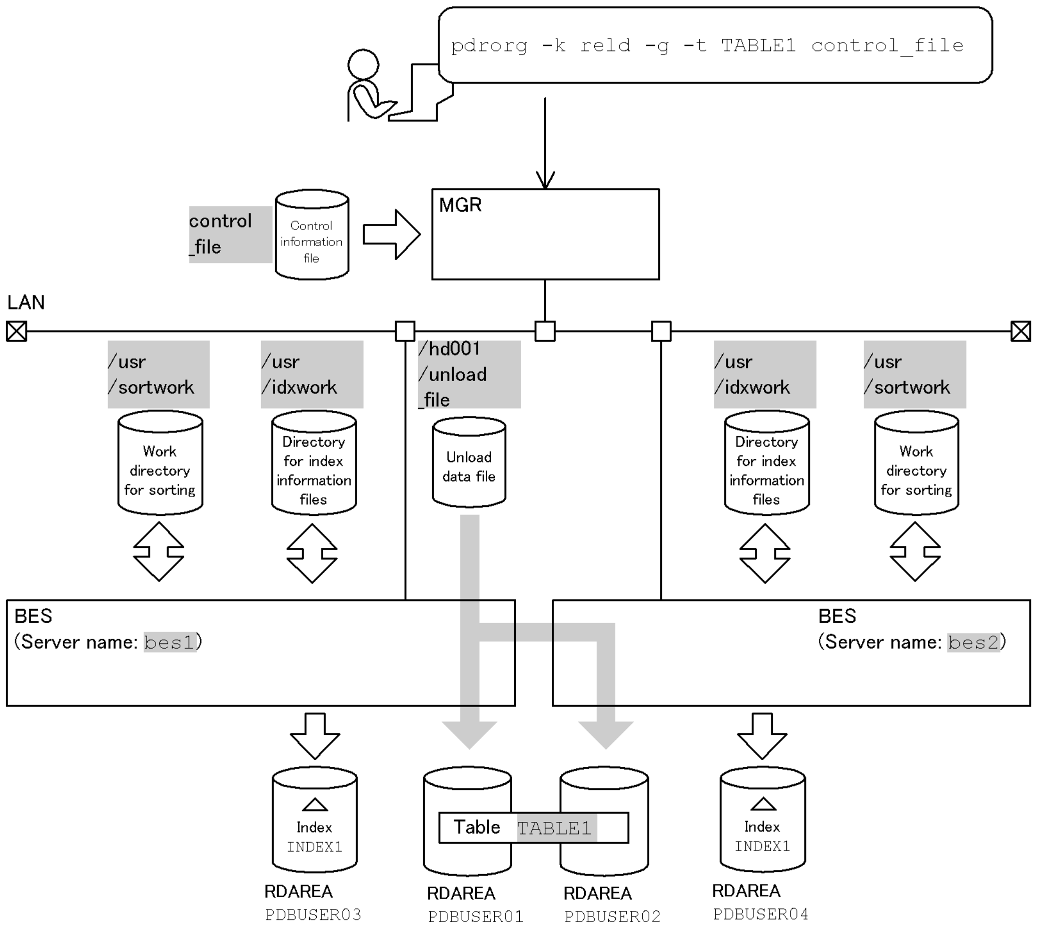
(b) Modifying the table's (TABLE1's) partitioning conditions

First, drop `TABLE1` with `DROP TABLE`. Then, redefine the table's (`TABLE1`'s) partitioning conditions and modify the index definition, shown as follows:

```
CREATE TABLE TABLE1(C1 INT NOT NULL,
                    C2 CHAR(8),
                    C3 INT)
             IN ((PDBUSER01) C1 > 10,(PDBUSER02))
CREATE INDEX INDEX1 ON TABLE1(C1)
             IN ((PDBUSER03),(PDBUSER04))
```

(c) Reloading to the table after modifying the partitioning conditions

Relationship between input/output files and RDAREAs



▨ Value specified in the database reorganization utility

Explanation of the command

The example reloads to the table (`TABLE1`).

`-k reld`: Specification for reloading

`-t TABLE1`: Name of the table being reloaded

`-g`: Specification for modifying the table partitioning conditions

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/hd001/unload_file                  1
idxwork bes1 /usr/idxwork                       2
idxwork bes2 /usr/idxwork                       2
sort bes1 /usr/sortwork,8192                    3
sort bes2 /usr/sortwork,8192                    3
```

Explanation:

1. Specifies the unload data file:

   `bes1`: Name of the server containing the unload data file

   `/hd001/unload_file`: Name of the unload data file

2. Specifies the directory for index information files to which index information is to be output:

   `bes1`, `bes2`: Names of the servers used to create index information files

   `/usr/idxwork`: Name of the directory for index information files

3. Specifies the work directory for sorting:

   `bes1`, `bes2`: Names of the servers used to create the sort work file

   `/usr/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)

(3) Unloading and reloading to migrate table data into another table

Example 3

This example migrates data from a table (`TABLE1)` to another table.

The example assumes that the following tables and indexes have been defined:

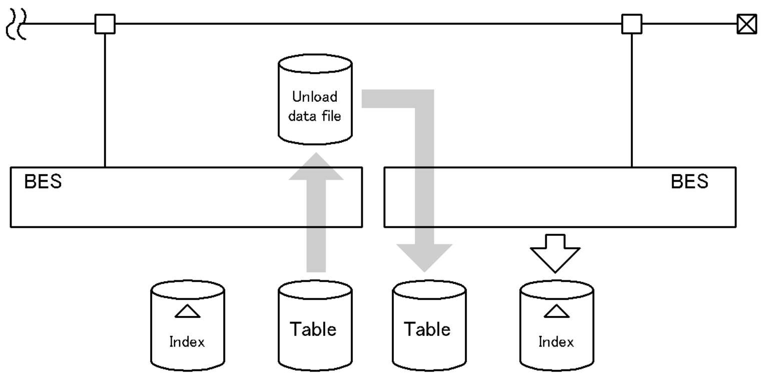
- Source table and index definition:

```
CREATE TABLE TABLE1(C1 INT NOT NULL,C2 CHAR(8),C3 INT) IN
PDBUSER01
CREATE INDEX INDEX1 ON TABLE1(C1) IN PDBUSER03
```

- Target table and index definition:

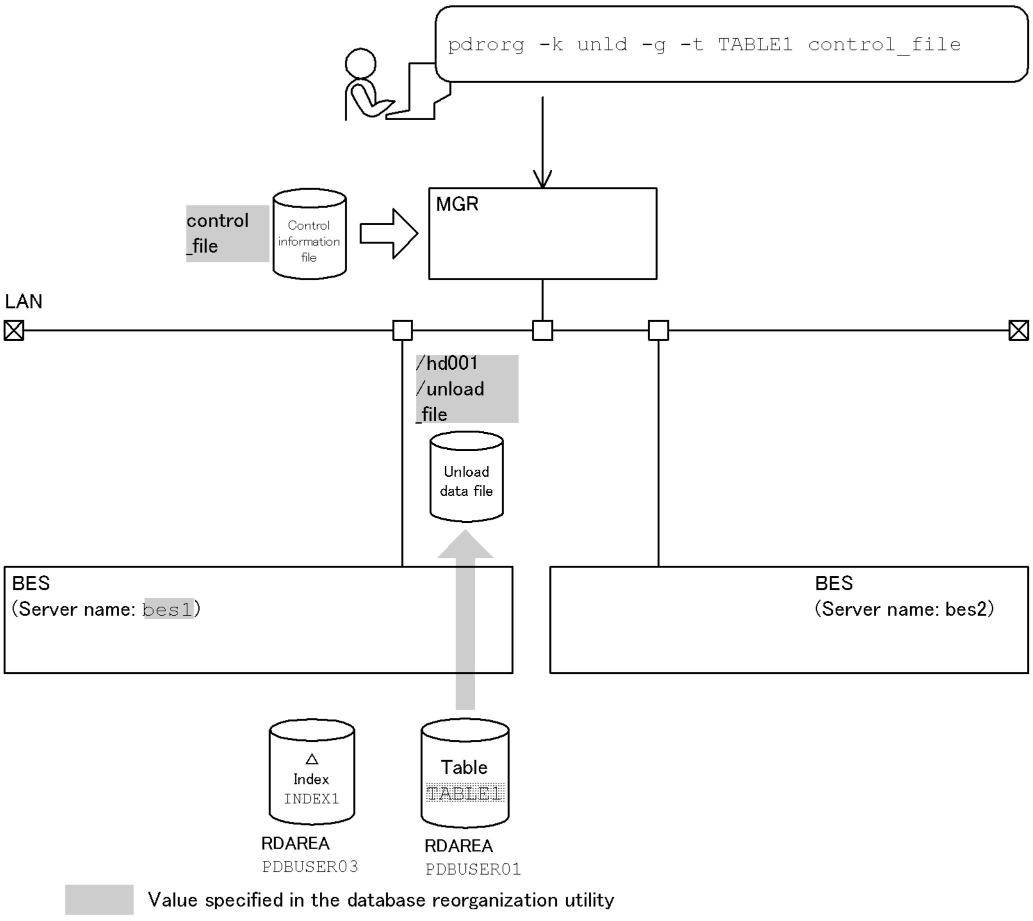
```
CREATE TABLE TABLE11(C1 INT NOT NULL,C2 CHAR(8),C3 INT) IN
PDBUSER11
CREATE INDEX INDEX11 ON TABLE11(C1) IN PDBUSER13
```

Overview



(a) Unloading the source table

Relationship between input/output files and RDAREAs



Explanation of the command

The example unloads the table (`TABLE1`).

`-k unld`: Specification for unloading

`-t TABLE1`: Name of the table being unloaded

`-g`: Specification for migrating data into another table

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/hd001/unload_file                              1
```

Explanation:

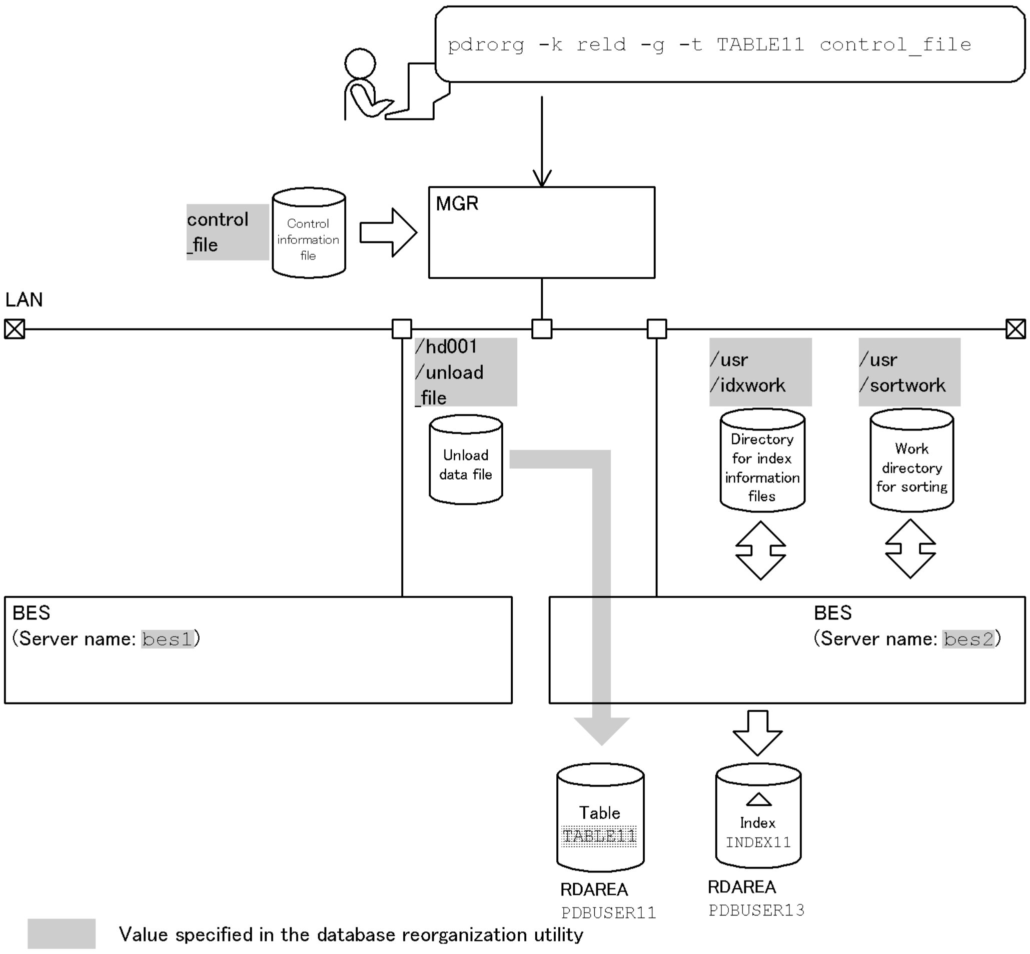
1. Specifies the unload data file:

   `bes1`: Name of the server containing the unload data file

   `/hd001/unload_file`: Name of the unload data file

(b) Reloading to the destination table (reloading to another table)

Relationship between input/output files and RDAREAs



Explanation of the command

The example reloads to the table (`TABLE11`).

`-k reld`: Specification for reloading

`-t TABLE11`: Name of the table being reloaded

`-g`: Specification for migrating data into another table

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/hd001/unload_file            1
idxwork bes2 /usr/idxwork                 2
sort bes2 /usr/sortwork,8192              3
tblname TABLE1                            4
```

Explanation:

1. Specifies the unload data file:

   `bes1`: Name of the server containing the unload data file

   `/hd001/unload_file`: Name of the unload data file

2. Specifies the directory for index information files to which index information is to be output:

   `bes2`: Name of the server used to create index information files

   `/usr/idxwork`: Name of the directory for index information files

3. Specifies the work directory for sorting:

   `bes2`: Name of the server used to create the sort work file

   `/usr/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)

4. Specifies reloading the table data into another table:

   `TABLE1`: Name of the source table

(4) Unloading and reloading while modifying the partitioning conditions of a table with an abstract data type

Example 4

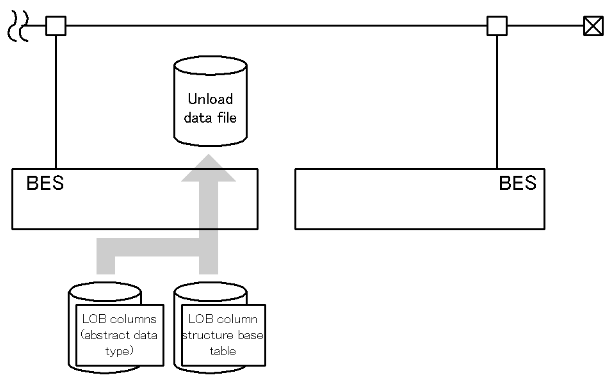
To modify the partitioning conditions of a table (`TABLE1`) with an abstract data type (`SGMLTEXT`), this example first unloads the table, modifies the table partitioning conditions, then reloads the table. The abstract data type (`SGMLTEXT`) is provided by the HiRDB Text Search Plug-in.

- Table definition:

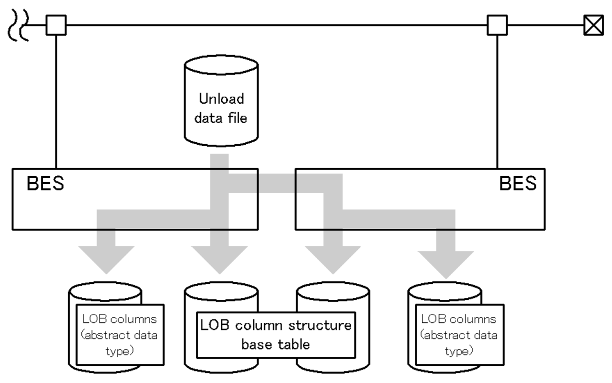
```
CREATE TABLE TABLE1(C1 INT,C2 SGMLTEXT ALLOCATE (SGMLTEXT IN
LOBUSER01)
                PLUGIN'<DTD>sgml.dtd</DTD>' ) IN PDUSER01
```

Overview

– **Unloading before modifying the table partitioning conditions**

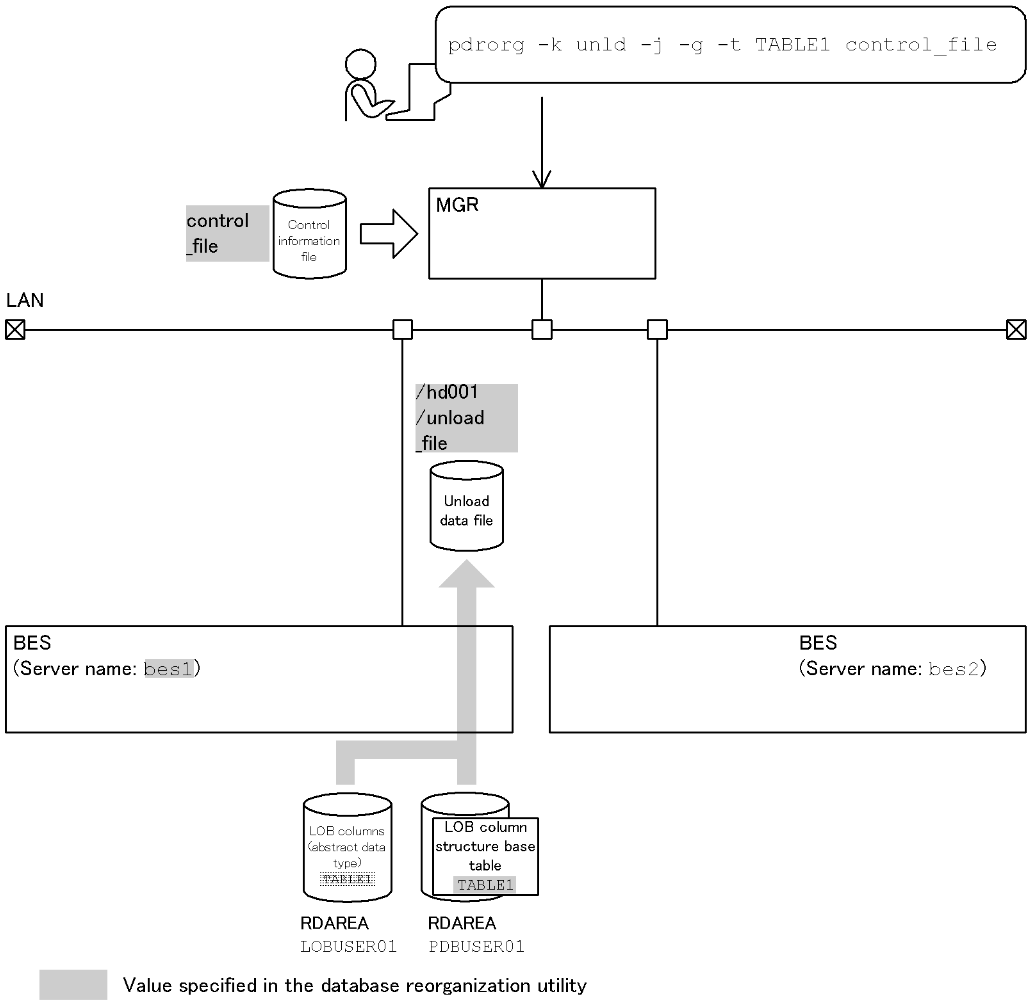

Unload data file
BES
BES
LOB columns (abstract data type)
LOB column structure base table

– **Reloading after modifying the table partitioning conditions**


Unload data file
BES
BES
LOB columns (abstract data type)
LOB column structure base table
LOB columns (abstract data type)

(a) Unloading the table before modifying the table partitioning conditions

Relationship between input/output files and RDAREAs


pdrorg -k unld -j -g -t TABLE1 control_file
control_file
Control information file
MGR
LAN
/hd001/unload_file
Unload data file
BES (Server name: bes1)
BES (Server name: bes2)
LOB columns (abstract data type) TABLE1
LOB column structure base table TABLE1
RDAREA LOBUSER01
RDAREA PDBUSER01
Value specified in the database reorganization utility

Explanation of the command

The example unloads the table (`TABLE1`).

`-k unld`: Specification for unloading

`-t TABLE1`: Name of the table being unloaded

`-j`: Specification for unloading a table with an abstract data type with LOB attribute

-g: Specification for modifying the table partitioning conditions

control_file: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/hd001/unload_file                            1
unld_func type=SGMLTEXT,func=unsgmltext(sgmltext)         2
```

Explanation:

1. Specifies the unload data file:

   bes1: Name of the server containing the unload data file

   /hd001/unload_file: Name of the unload data file

2. Specifies the constructor parameter reverse creation function:

   SGMLTEXT: Name of the abstract data type (column C2)

   unsgmltext: Name of the constructor parameter reverse creation function (for the actual name, see the applicable plug-in manual)

   sgmltext: Data type of the argument of the constructor parameter reverse creation function

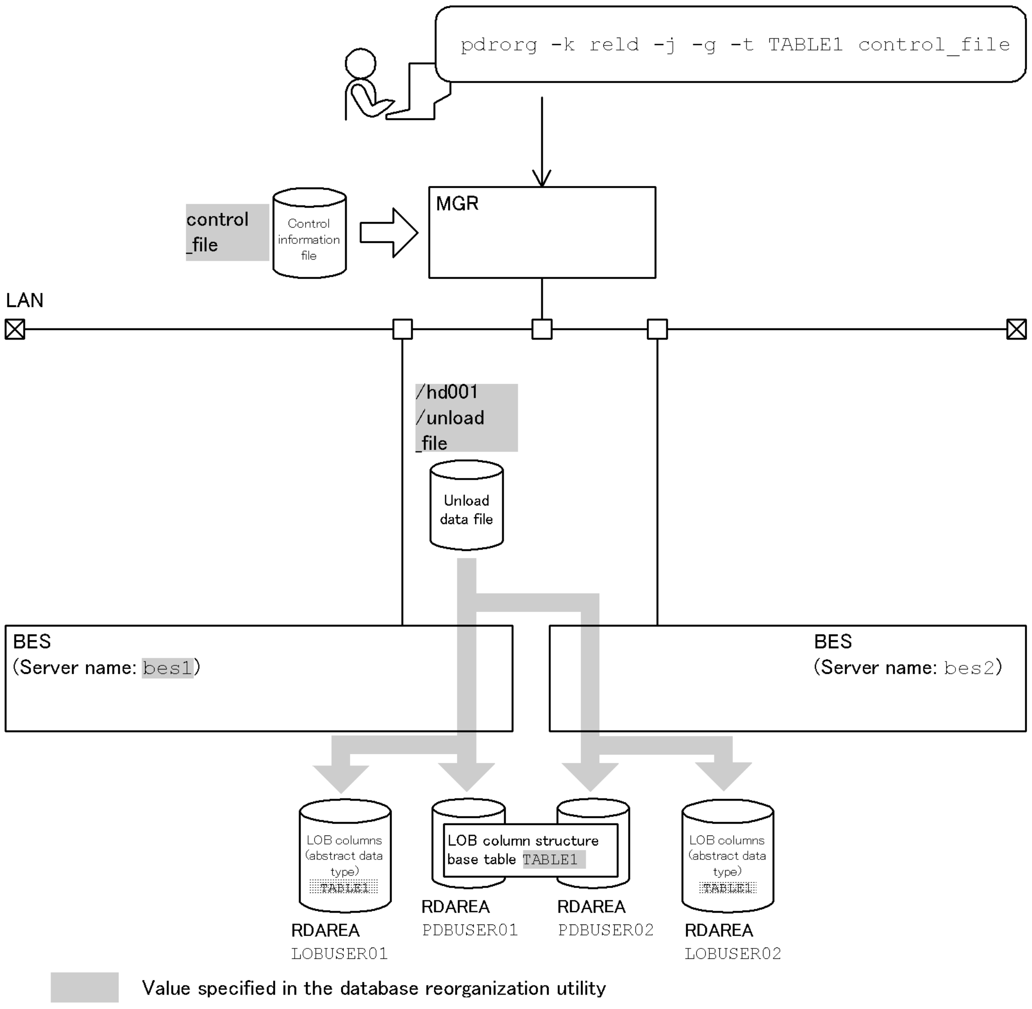
(b) Modifying the table's (TABLE1's) partitioning conditions

First, drop TABLE1 with DROP TABLE. Then, redefine the table's (TABLE1's) partitioning conditions as follows:

```
CREATE TABLE TABLE1(C1 INT NOT NULL,
                    C2 SGMLTEXT ALLOCATE
                        (SGMLTEXT IN ((LOBUSER01),(LOBUSER02)))
                        PLUGIN'<DTD>sgml.dtd</DTD>' )
   IN ((PDUSER01) C1<2000,(PDBUSER02))
```

(c) Reloading to the table after modifying the partitioning conditions

Relationship between input/output files and RDAREAs



Explanation of the command

The example reloads to the table (`TABLE1`).

`-k reld`: Specification for reloading

`-t TABLE1`: Name of the table being reloaded

`-j`: Specification for reloading to a table with an abstract data type with LOB attribute

`-g`: Specification for modifying the table partitioning conditions

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/hd001/unload_file                    1
reld_func type=SGMLTEXT,func=sgmltext(blob)       2
```

Explanation:

1. Specifies the unload data file:

   `bes1`: Name of the server containing the unload data file

   `/hd001/unload_file`: Name of the unload data file

2. Specifies the constructor function:

   `SGMLTEXT`: Name of the abstract data type (column `C2`)

   `sgmltext(blob)`: Name of the constructor function (parentheses enclose the data type of the constructor function's argument)

(5) Unloading and reloading to migrate table data from a table with an abstract data type to another table

Example 5

This example migrates data from a table (`TABLE1)` with abstract data type (`SGMLTEXT`) to another table. The abstract data type (`SGMLTEXT`) is provided by the HiRDB Text Search Plug-in.

- Source table and index definition:

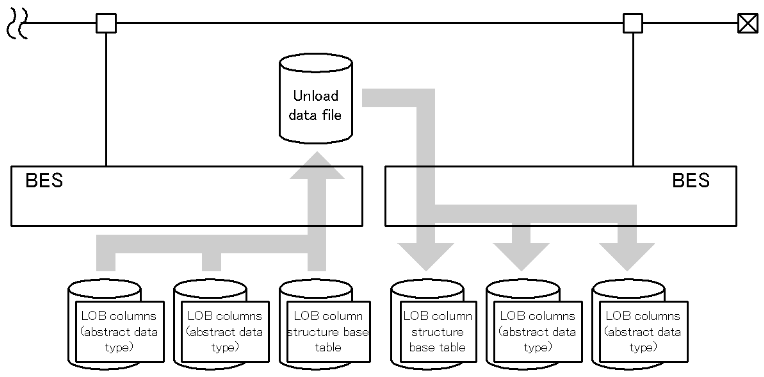
```
CREATE TABLE TABLE1(C1 INT,
                    C2 SGMLTEXT ALLOCATE
                      (SGMLTEXT IN LOBUSER01)
                      PLUGIN'<DTD>sgml.dtd</DTD>'
                    C3 SGMLTEXT ALLOCATE
                      (SGMLTEXT IN LOBUSER02)
                      PLUGIN'<DTD>sgml.dtd</DTD>'
                    ) IN PDUSER01
```

- Target table and index definition:

```
CREATE TABLE TABLE2(C1 INT,
                    C2 SGMLTEXT ALLOCATE
                      (SGMLTEXT IN LOBUSER03)
                      PLUGIN'<DTD>sgml.dtd</DTD>'
                    C3 SGMLTEXT ALLOCATE
                      (SGMLTEXT IN LOBUSER04)
                      PLUGIN'<DTD>sgml.dtd</DTD>'
```

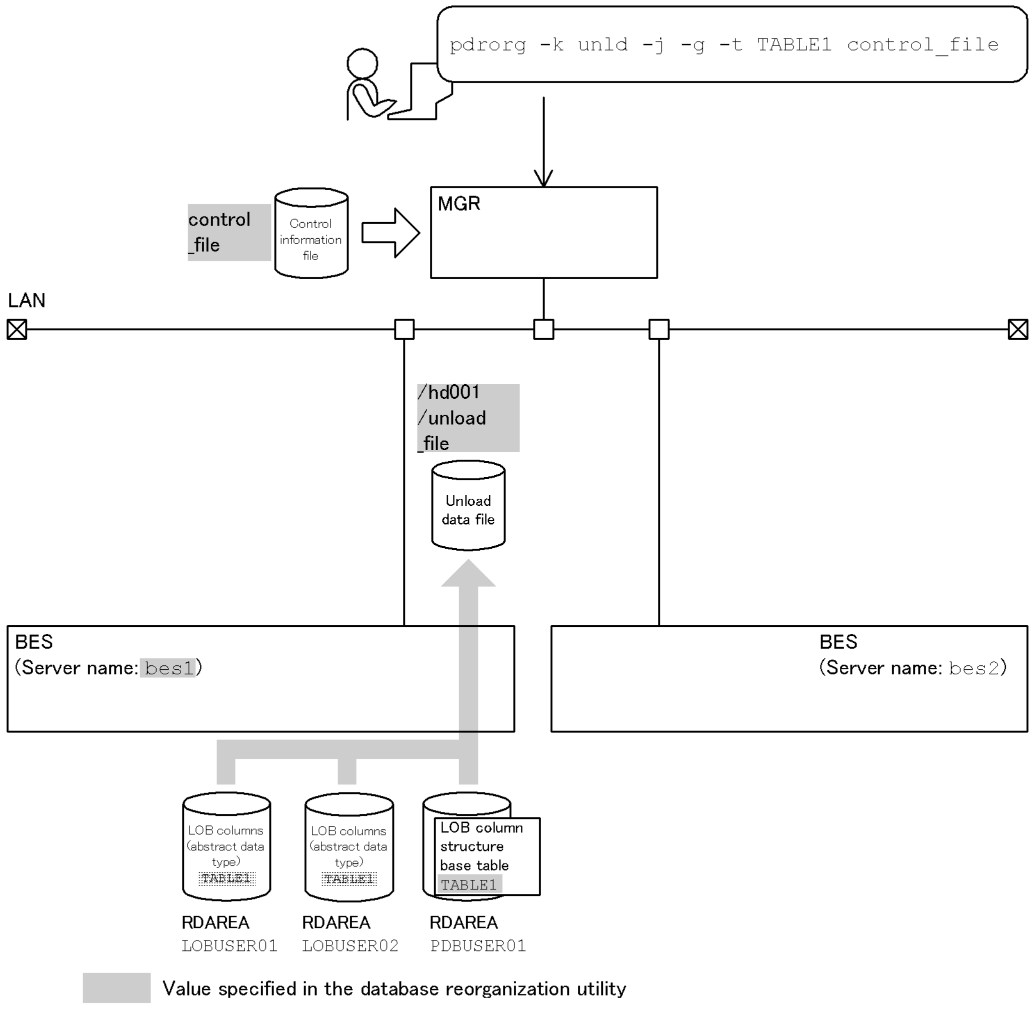
```
                ) IN PDUSER02
```

Overview


Unload data file
BES
BES
LOB columns (abstract data type)
LOB columns (abstract data type)
LOB column structure base table
LOB column structure base table
LOB columns (abstract data type)
LOB columns (abstract data type)

(a) Unloading the source table

Relationship between input/output files and RDAREAs


pdrorg -k unld -j -g -t TABLE1 control_file
control_file
Control information file
MGR
LAN
/hd001/unload_file
Unload data file
BES (Server name: bes1)
BES (Server name: bes2)
LOB columns (abstract data type) TABLE1
LOB columns (abstract data type) TABLE1
LOB column structure base table TABLE1
RDAREA LOBUSER01
RDAREA LOBUSER02
RDAREA PDBUSER01
Value specified in the database reorganization utility

Explanation of the command

The example unloads the table (`TABLE1`).

`-k unld`: Specification for unloading

`-t TABLE1`: Name of the table being unloaded

`-j`: Specification for unloading a table with an abstract data type with LOB attribute

`-g`: Specification for migrating data into another table

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/hd001/unload_file                          1
unld_func type=SGMLTEXT,func=unsgmltext(sgmltext)       2
```

Explanation:

1. Specifies the unload data file:

   `bes1`: Name of the server containing the unload data file

   `/hd001/unload_file`: Name of the unload data file

2. Specifies the constructor parameter reverse creation function:

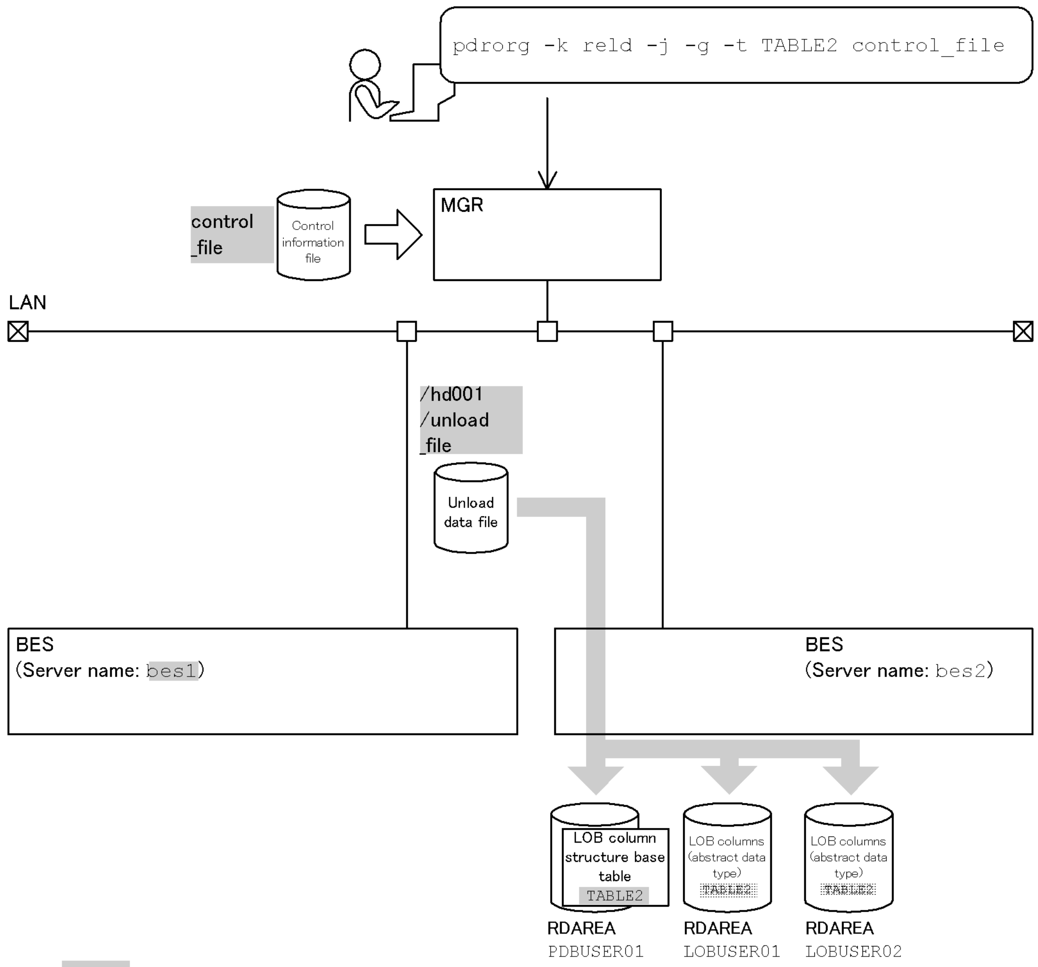
   `SGMLTEXT`: Name of the abstract data type (column `C2`)

   `unsgmltext`: Name of the constructor parameter reverse creation function (for the actual name, see the applicable plug-in manual)

   `sgmltext`: Data type of the argument of the constructor parameter reverse creation function

(b) Reloading to the target table (another table)

Relationship between input/output files and RDAREAs



▨ Value specified in the database reorganization utility

Explanation of the command

The example reloads to the table (`TABLE2`).

`-k reld`: Specification for reloading

`-t TABLE2`: Name of the table being reloaded

`-j`: Specification for reloading to a table with an abstract data type with LOB attribute

`-g`: Specification for migrating data to another table

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/hd001/unload_file                          1
reld_func type=SGMLTEXT,func=sgmltext(blob)             2
tblname TABLE1                                          3
```

Explanation:

1. Specifies the unload data file:

   `bes1`: Name of the server containing the unload data file

   `/hd001/unload_file`: Name of the unload data file

2. Specifies the constructor function:

   `SGMLTEXT`: Name of the abstract data type (column `C2`)

   `sgmltext(blob)`: Name of the constructor function (parentheses enclose the data type of the constructor function's argument)

3. Specifies reloading table data to another table:

   `TABLE1`: Name of the source table

(6) Creating an input data file for pdload in fixed-size data format

Example 6

This example unloads data from the table `PRODUCT_TABLE` as an input data file for `pdload` in fixed-size data format.

- Table definition

```
  CREATE TABLE PRODUCT_TABLE
         (PRODUCT_NUMBER SMALLINT NOT NULL,PRODUCT_NAME
NVARCHAR(12),QUANTITY INTEGER)
         IN
((RDAREA1)PRODUCT_NUMBER<=10000,(RDAREA2)PRODUCT_NUMBER<=20
000,(RDAREA3));
```

pdrorg command

```
pdrorg -k unld -t PRODUCT_TABLE -W fixtext,@,cr control_file
```

Explanation

`-k unld`: Specification for unloading

`-t PRODUCT_TABLE`: Name of the table being unloaded

`-W fixtext,@,cr`: Specification for unloading the data as an input data file for `pdload` in fixed-size data format (`@` is the padding character, and use of 1-byte linefeed codes is specified)

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload bes1:/hd001/unload_file                                        1
fixtext_option enclose=" format=integer,type1 format=smallint,type2   2
```

Explanation:

1. Specifies the unload data file:

   `bes1`: Name of the server containing the unload data file

   `/hd001/unload_file`: Name of the unload data file

2. Specifies how to edit the output data:

   `enclose="`: Encloses in the enclosing character (`"`) the data for `PRODUCT_NAME` in a column whose data type is `NVARCHAR`.

   `format=integer,type1`: Outputs in output format type1 the data for

`QUANTITY` in a column whose data type is `INTEGER`.

`format=smallint,type2`: Outputs in output format type 2 the data for `PRODUCT_NUMBER` in a column whose data type is `SMALLINT`.

Output data

```
PRODUCT_NUMBER    PRODUCT_NAME    QUANTITY
(6 bytes)         (14 bytes)      (11 bytes)     (1 byte)

###100            "BOOTS"@@@@@@@  @@@@@@@@@@@    linefeed-character
@@@@@@            ""@@@@@@@@@@@@  #0000000100    linefeed-character
-12345            @@@@@@@@@@@@@@  -0000000300    linefeed-character
```

Note

# represents a space character, and 1 row consists of 32 bytes.

8.3.2 Cross-reference by purpose

Required options and control statements depend on the type of unload operation executed by the user.

The options listed in (1) and the control statements are mandatory. The items in (2) through (4) are related to the options and control statements presented in the *Reference* column. For details about the options, see Section 8.9.2.

(1) Required items

Item	Reference			
	Option		Control statement	
Type of `pdrorg` processing	`-k unld`	(1)	—	—
Reorganization of a user-defined table (can be omitted because the default is `-c user`)	`-c user`	(2)	—	—
Name of the table to be unloaded	`-t`	(3)	—	—
File containing the control statements	*control-information-filename*	(19)	—	—
Information about an unload data file	—	—	`unload` statement	8.9.4

— : Not applicable

(2) Information to be specified depending on the attribute of the table being unloaded

Item	Reference location			
	Option		Control statement	
There are LOB columns	`-j`	(9)	`lobunld` statement	8.9.9
There are columns of abstract data type provided by a plug-in (LOB attribute)	`-j`	(9)	`unld_func` statement	8.9.13
There are repetition columns	`-W`	(7)	`array` statement	8.9.12
Consecutive trailing spaces are to be removed from a table with suppress option specified*	`-S`	(14)	—	—
Inner replica facility is used	`-q`	(18)	—	—

—: Not applicable

* This option helps you reduce the processing time and the size of the unload data file.

(3) Information to be specified depending on the type of unload data file or LOB data unload file

Item	Reference location			
	Option		Control statement	
Using unload data as `pdload`'s input data file or with a UAP	`-W`	(7)	`fixtext_option` statement	8.9.19
Integrating data into a single file	`-g`	(8)	—	—
Using EasyMT	`-f`	(12)	`mtguide` statement	8.9.3
			`emtdef` statement	8.9.3
Using HiRDB files	`-f`	(12)	—	—
Specifying the unload data storage sequence (in order of cluster key, index, or data storage)	`-b`	(13)	—	—

—: Not applicable

(4) Information to be specified depending on the unloading method

Item	Reference location			
	Option		Control statement	
Unloading only some of the RDAREAs for a row-partitioned table	`-r`	(4)	—	—
Changing the `pdrorg` executor's user authorization identifier to a value other than the value of environment variable `PDUSER`[1]	`-u`	(10)	—	—
Unloading in batch input/output mode using local buffer instead of global buffer[2]	`-n`	(11)	—	—
Changing an unloading status message output interval to a value other than 100,000 lines	`-m`	(16)	—	—
Monitoring the response time for server-to-server communication	-X	(17)	—	—
Monitoring the execution time of `pdrorg`	—	—	`option` statement	8.9.16
Using the same space character in the entire table data	—	—	`option` statement	8.9.16
Obtaining tuning information during the execution of `pdrorg`	—	—	`report` statement	8.9.17
Unloading a column of abstract data type with `BLOB` attribute as `VARCHAR` type	—	—	`blobtovarchar` statement	8.9.18
Executing one of the following: • Deleting unneeded data and then outputting the unload data file • Updating data to use as `pdload`'s input data file • Outputting data in a desired format for applications	—	—	`unlduoc` statement	8.9.10

— : Not applicable

[1] If omitted, the system assumes the value of the `PDUSER` environment variable. If the `PDUSER` environment variable has not been specified, the system assumes the user name in the login window.

[2] By specifying the number of batch input/output pages, you can reduce the number of I/O operations, because data is input/output in units of specified pages in batch mode.

8.3.3 Format of database load utility input files

If you specify the `-W` option at the time of unloading, you can use the unload data file as the input data file to the database load utility.

(1) DAT and extended DAT formats

In the DAT and extended DAT formats, data is output as character string data. For details about the differences between the DAT and extended DAT formats, see *(c) Differences between DAT and extended DAT formats*. Except for the cited differences, the DAT and extended DAT formats are identical.

To output data, use the following rules:

1. The system outputs one table row as one line in the file (a linefeed character is output at the end of each line).
2. The system inserts a separator character between column data. For details about column data output formats (in DAT format) for various data types, see Section *5.5.1 DAT format*.

   Character string, national character, mixed character string, and `BINARY` data are enclosed in double quotation marks (`"`) in the output format.

   The output format of an abstract data type consists of as many data items as there are specified by the constructor parameter reverse creation function, each of which is delimited by a separator character. If the constructor parameter reverse creation function's return value is BLOB, the system outputs consecutive separator characters as the null value.
3. The system outputs LOB column data as the null value, regardless of the specification of the `-j` option.
4. If the constructor parameter reverse creation function's return value is BLOB, the system outputs the column data of abstract data type as the null value.
5. An error results if the line length exceeds 512 MB after conversion. If you assign a streaming tape device to the unload data file and the line length exceeds 32 KB, an error results.
6. If you specify the `spacelvl=3` and `sup` options in the `option` statement, the system converts two consecutive single-byte spaces of `NCHAR` type and suppresses the output of a pair of single-byte spaces.
7. If the data type of a column is character string, national character data, mixed character string, or binary data, and the column value contains a combination of the enclosing character (`"`) + separator character or the separator character + enclosing character (`"`), the utility may treat the value as the end or start of column data during data loading. To avoid this, use the `-W` option to change the separator character during unloading or output the data in the binary format.

(a) Format for repetition columns

For repetition columns, the system outputs data from each element by separating it with separator characters. The row data format depends on what is specified in the `array` statement in the control information file. Figure 8-13 shows the row data output format when `ff` is specified in the `array` statement: Figure 8-14 shows the row data output format when `vv` is specified in the `array` statement.

Figure 8-13: Row data output format with ff specified in the array statement

Example: Column with `CHAR(1) ARRAY[10]`

```
a, b, c, d, e, f, g, h, i, j        1
a, b, c, d, , f, , , ,              2
a, b, , d, e, f, g, , , j           3
, , , , , , , , ,                   4
```

Explanation

1. Data is stored in all the elements.
2. Elements 1-6 contain data, and element 5 contains a null value. Elements 7-10 contain either no data or a null value.
3. Data is stored in all the elements. Elements 3, 8, and 9 contain a null value.
4. Either the column value is null or all elements contain a null value.

Figure 8-14: Row data output format with vv specified in the array statement

Example: Column with `CHAR(1) ARRAY[10]`

```
10, a, b, c, d, e, f, g, h, i, j    1
6, a, b, c, d, , f                  2
10, a, b, , d, e, f, g, , ,         3
0                                   4
10, , , , , , , , , ,               5
```

Explanation

1. Data is stored in all the elements. A value of `10` is output as the current number of elements.
2. Elements 1-6 contain data, and element 5 contains a null value. A value of `6` is output as the current number of elements.
3. Data is stored in all the elements. Elements 3 and 8-10 contain null values. A value of `10` is output as the current number of elements.
4. Because a null value is stored in the column value, a value of `0` is output as the current number of elements.

5. All elements contain a null value. A value of `10` is output as the current number of elements.

(b) Format used when the sup option is specified

If you specify the `sup` option during unload operation, the system compresses any trailing spaces in any column that is shorter than the column length specified in the table definitions and then outputs them to the unload data file. The following shows the format depending on the specification of the `sup` option:

Data type: CHAR or MCHAR

Example 1

- Table definition:

```
CREATE TABLE T1(C1 INTEGER,C2 CHAR(10),...);
```

Input data: `1,A,A,...`

`sup` option: Specified

```
1,"AA",...
```

Explanation: The system compresses the spaces (equal to eight characters) in a column that is shorter than the defined length.

`sup` option: Not specified

```
1,"AA........",...
```

Note: A period (`.`) indicates a single-byte space.

Explanation: The system outputs single-byte spaces (`.`) (equal to eight characters) to pad the column that is shorter than the defined length.

Example 2

- Table definition:

```
CREATE TABLE T1(C1 INTEGER,C2 MCHAR(10),...1,);
```

Input data: `1,..........,...` (a period (`.`) indicates a single-byte space)

`sup` option: Specified

```
1,".",...
```

Explanation: The system outputs one single-byte space.

`sup` option: Not specified

```
1,"..........",...
```

Explanation: The system outputs as many single-byte spaces as there are defined for the table.

Data type: NCHAR

Example 1

- Table definition:

```
CREATE TABLE T1(C1 INTEGER,C2 NCHAR(10),...);
```

Input data: `1,A,A,...`

`sup` option: Specified

```
1,"AA",...
```

Explanation: The system compresses the spaces (equal to eight characters) in a column that is shorter than the defined length.

`sup` option: Not specified

```
1,"AA∆ ∆ ∆ ∆ ∆ ∆ ∆ ∆",...
```

Note: A triangle (∆) indicates a double-byte space.

Explanation: The system outputs double-byte spaces (∆) (equal to eight characters) to pad the column that is shorter than the defined length.

Example 2

- Table definition:

```
CREATE TABLE T1(C1 INTEGER,C2 NCHAR(10),...1,);
```

Input data: `1,∆∆∆∆∆∆∆∆∆∆,...` (a triangle (∆) indicates a single-byte space)

`sup` option: Specified

```
1,"∆",...
```

Explanation: The system outputs one double-byte space.

`sup` option: Not specified

```
1,"∆ ∆ ∆ ∆ ∆ ∆ ∆ ∆ ∆ ∆",...
```

Explanation: The system outputs as many double-byte spaces as there are defined for the table.

Data type: NCHAR (spacelvl=3 specified in the option statement)

Example 1

- Table definition:

```
CREATE TABLE T1(C1 INTEGER,C2 NCHAR(10),...);
```

Input data: `1,A,A,...`

`sup` option: Specified

```
1,"AA",...
```

Explanation: The system compresses the spaces (equal to eight characters) in a column that is shorter than the defined length.

`sup` option: Not specified

```
1,"AA................",...
```

Explanation: The system outputs single-byte spaces (equal to eight characters) to pad the column that is shorter than the defined length.

Example 2

- Table definition:

```
CREATE TABLE T1(C1 INTEGER,C2 NCHAR(10),...1,);
```

Input data: `1,△ △ △ △ △ △ △ △ △ △,...` (a triangle (△) indicates a single-byte space)

`sup` option: Specified

```
1,"..",...
```

Note: A period (`.`) indicates a single-byte space.

Explanation: The system outputs two single-byte spaces.

`sup` option: Not specified

```
1,"....................",...
```

Note: A period (`.`) indicates a single-byte space.

Explanation: The system outputs as many single-byte spaces as there are defined for the table.

(c) Differences between DAT and extended DAT formats

The following table describes the differences between the DAT and extended DAT formats:

Item	DAT format	Extended DAT format
When the data contains the null character (`\0`) or linefeed character (`\n`)	The row containing the corresponding column data value is not output.	The row containing the corresponding column data value is output (`\0` and `\n` are output as is).
When the enclosing character (`"`) is used as part of the data	The enclosing character used as part of the corresponding column data value is output as is.	Two consecutive enclosing characters are output for an enclosing character that is used as part of the column data value.* Example: `"` ➔ `""`

* Because multi-byte characters are also subject to conversion, you must pay special attention if the file contents are to be referenced, such as by an application program. Data loading can be performed because `pdload` converts `""` to `"` whether or not the data contains multi-byte characters.

(2) Binary format

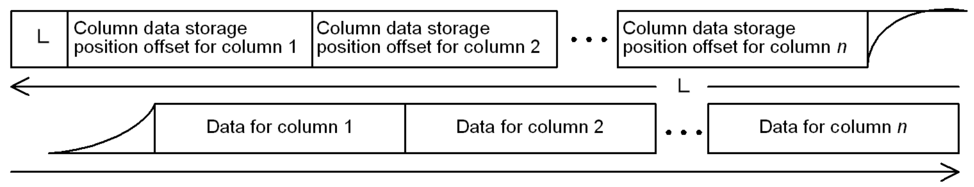
In the binary format, data is output in an internal format based on the database column definition. If this type of file is used as an input to the database load utility, the unloaded table and the data loading target table must have the same column definitions. For a non-FIX table, data cannot be loaded into a FIX table, even if the column definitions are the same.

To output data, use the following rules:

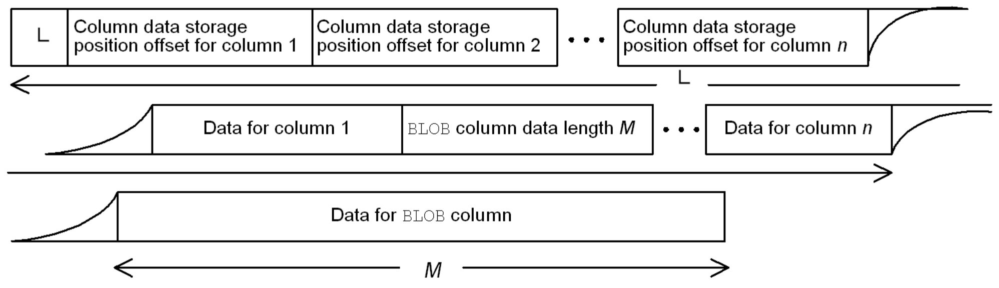
1. For FIX tables, the system outputs data continuously without inserting separator characters between the column data items. For non-FIX tables, the system outputs not only the column data but also the data length and data position offset. Figure 8-15 shows the row data output format for non-FIX tables. For a description of column data output formats (in binary format) for various data types, see Section *5.5.2 Binary format*.
2. For a table with LOB columns defined, if the `-j` option is specified, the system outputs the LOB data value; otherwise, the system outputs the null value.
3. In the case of data output to a fixed-length block tape such as CMT, the system attaches a 512-byte header to the data.

Figure 8-15: Row data output format for non-fix tables

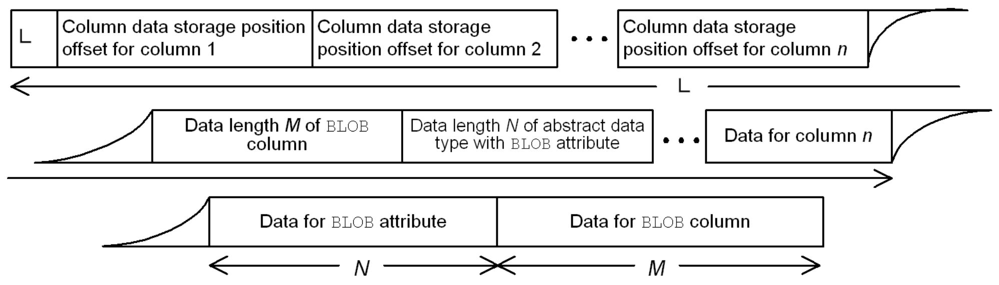
- When all columns have defined types


L
Column data storage position offset for column 1
Column data storage position offset for column 2
Column data storage position offset for column *n*
L
Data for column 1
Data for column 2
Data for column *n*

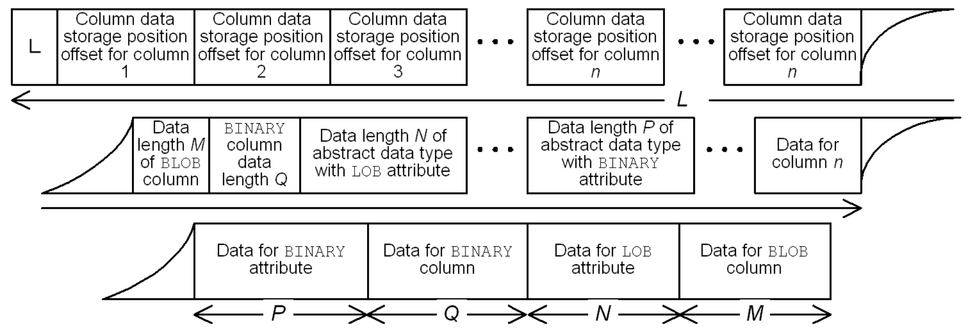
- When column 2 is `BLOB` type


L
Column data storage position offset for column 1
Column data storage position offset for column 2
Column data storage position offset for column *n*
L
Data for column 1
BLOB column data length *M*
Data for column *n*
Data for BLOB column
M

- When column 1 is `BLOB` type and column 2 is abstract data type with `BLOB` attribute


L
Column data storage position offset for column 1
Column data storage position offset for column 2
Column data storage position offset for column *n*
L
Data length *M* of BLOB column
Data length *N* of abstract data type with BLOB attribute
Data for column *n*
Data for BLOB attribute
Data for BLOB column
N
M

- When column 1 is `BLOB` type, column 2 is `BINARY` type, and column 3 is abstract data type with `BLOB` attribute


L
Column data storage position offset for column 1
Column data storage position offset for column 2
Column data storage position offset for column 3
Column data storage position offset for column *n*
Column data storage position offset for column *n*
L
Data length *M* of BLOB column
BINARY column data length *Q*
Data length *N* of abstract data type with LOB attribute
Data length *P* of abstract data type with BINARY attribute
Data for column *n*
Data for BINARY attribute
Data for BINARY column
Data for LOB attribute
Data for BLOB column
P
Q
N
M

Explanation

The system outputs the row length, column data offset, and row data in this order.

L

Row length (4-byte binary number)

Column data storage position offset

Offset from the beginning of the line to the column data header (4-byte binary number). The system outputs as many column data offsets as there are columns in the table. If a column's data is the null value, the system stores a value of 0 and does not output column data.

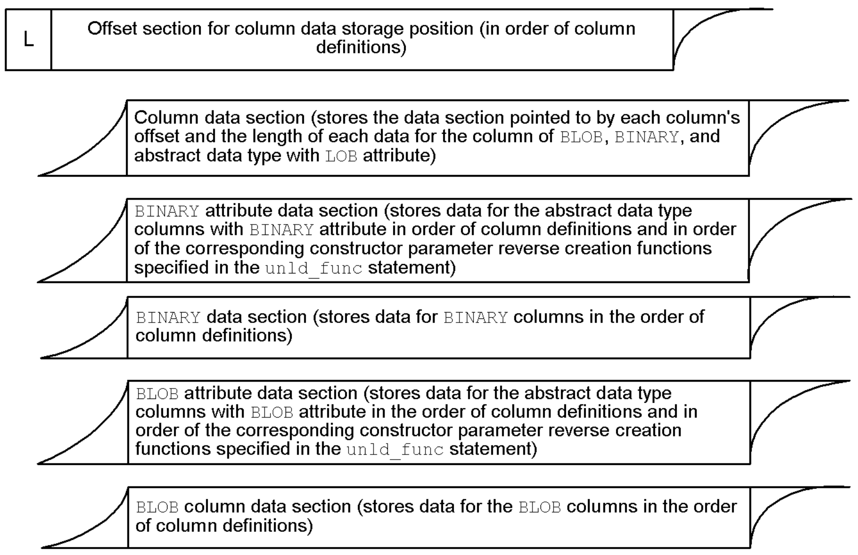
Column data

Column value.

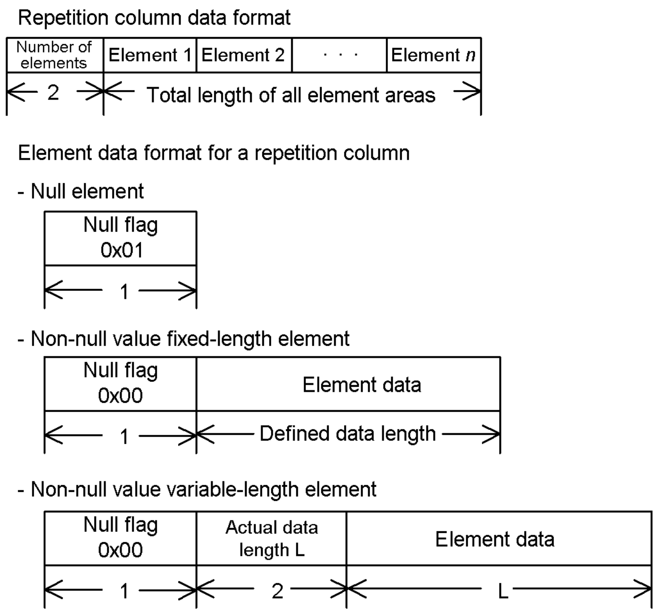
If the column data requires a word boundary to be referenced by a UAP, first copy it to a boundary-adjusted area, then reference the column data.

For details about the output format for columns whose column data type or abstract data type attribute is `BINARY`, see *5.5.2 Binary format*.

The following shows the order following the `LOB` column structure base table of the `BINARY` data, `BINARY` attribute data of abstract data type, `LOB` attribute data of abstract data type, and `LOB` column data:

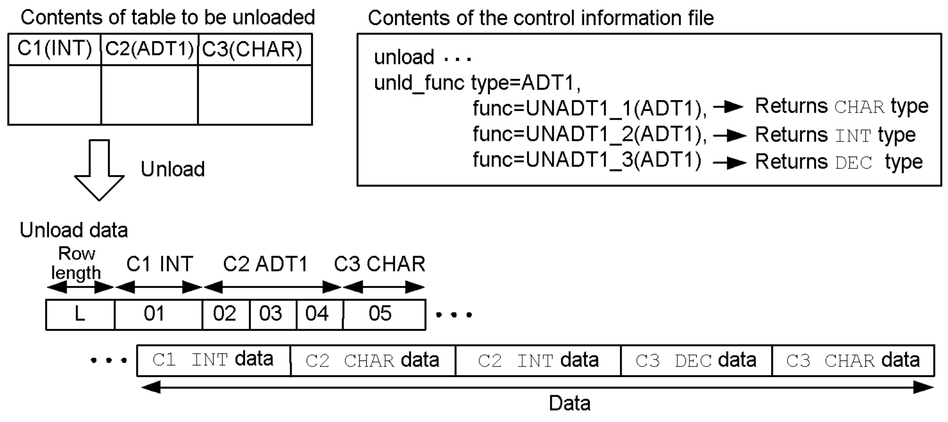

L
Offset section for column data storage position (in order of column definitions)
Column data section (stores the data section pointed to by each column's offset and the length of each data for the column of BLOB, BINARY, and abstract data type with LOB attribute)
BINARY attribute data section (stores data for the abstract data type columns with BINARY attribute in order of column definitions and in order of the corresponding constructor parameter reverse creation functions specified in the unld_func statement)
BINARY data section (stores data for BINARY columns in the order of column definitions)
BLOB attribute data section (stores data for the abstract data type columns with BLOB attribute in the order of column definitions and in order of the corresponding constructor parameter reverse creation functions specified in the unld_func statement)
BLOB column data section (stores data for the BLOB columns in the order of column definitions)

Repetition columns take the following format:


Repetition column data format
Number of elements
Element 1
Element 2
· · ·
Element *n*
2
Total length of all element areas
Element data format for a repetition column
- Null element
Null flag 0x01
1
- Non-null value fixed-length element
Null flag 0x00
Element data
1
Defined data length
- Non-null value variable-length element
Null flag 0x00
Actual data length L
Element data
1
2
L

Column data storage position offset (abstract data type with `BLOB` attribute)

The system outputs as many data offsets as there are specified by the constructor parameter reverse creation function in the specified order. Therefore, the number of columns in an unloaded table may not match the number of offsets in the unloaded data. The following shows an example:



LOB attribute data

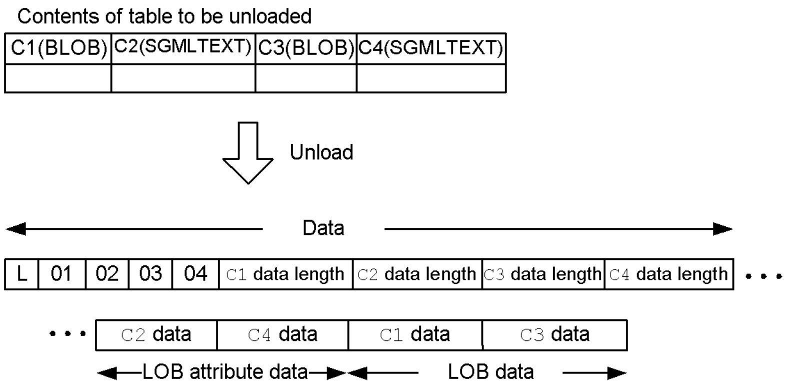
The system outputs the data returned from the constructor parameter reverse creation function in the same format as for a column of defined type. The order of output is the same as the order of offsets, which is the order specified by the constructor parameter reverse creation function.

In the data indicated by row length *L*, the system outputs two-byte control information for each column of abstract data type. There is no need to consider this control information to edit the data.

If the constructor parameter reverse creation function returns a `BLOB` type, the system outputs the data in the same manner as with a defined `BLOB` type in the following order:

1. The system outputs the column data of defined types and the data of abstract data type that is output by the constructor parameter reverse creation function (for a `BLOB` type, only the data length) as the LOB column structure base table data with a length of *L*.
2. Immediately after the LOB column structure base table data, the system outputs the `BLOB` data that is output by the constructor parameter reverse creation function. The order of output is the same as the order defined for the column of abstract data type. If multiple constructor parameter reverse creation functions have been specified for this abstract data type, the system outputs the data in the order the functions are specified.
3. The system outputs all other `BLOB` data that is not an abstract data type. The order of output is the same as the order in which the `BLOB` column is defined.

The following shows an example:



(3) Fixed-size data format

In the fixed-size data format, data is output as fixed-length character string data.

(a) Output format

- Output unit

  One row of data is output per row in the database.
- Output format of each row
  - Length of a row

    A row is output as fixed length.
  - Output order of column data

    Column data is output in the order of the column definitions.
  - Separator between column items

    No separator character is output between column data items.
  - Linefeed

    By specifying the applicable option, you can insert a linefeed character at the end of the row.
- Output format of each column

  For details about the output format of each type of column data, see Table 8-3.
- Output format of repetition columns

  Data for each element is output as many times as defined for the element. For details about the output format of each type of element data, see Table 8-3.
- Output format of special column data
  - Column data with the null value

    The column data is padded with the padding character up to the output length of the column data. Repetition columns are padded with as many instances of the padding character as equals (output length of column data × number of defined elements).
  - Repetition column whose element has the null value

    Column data is padded with the padding character up to the output length of the column data.
  - Control characters in column data

    For the character data, mixed character string data, and national character data types, you can use the `fixtext_option` statement to specify how to handle control characters. The default is that the utility does not check for control characters.

Table 8-3: Output format of each type of column data

Data type of column		**Output format**	**Output length (bytes)**
Numeric data	`INTEGER`	• When `format=integer,type1` is specified in the `fixtext_option` statement or the `fixtext_option` statement is omitted - A numeric value is right-aligned. - The first byte is treated as the sign part: the space character (`0x20`) is output for a positive value and - is output for a negative value. - An empty part is filled with `0`. Example: Δ`0000000001` `-0000000001` Δ: One space character • When `format=integer,type2` is specified in the `fixtext_option` statement - A numeric value is right-aligned. - The leading digit of the numeric value is treated as the sign part: the space character (`0x20`) is output for a positive value and - is output for a negative value. - An empty part is filled with space characters (`0x20`). Example: Δ Δ Δ Δ Δ Δ Δ Δ Δ Δ`2` Δ Δ Δ Δ Δ Δ Δ Δ Δ`-2` Δ: One space character	11

Data type of column		Output format	Output length (bytes)

	`SMALLINT`	• When `format=smallint,type1` is specified in the `fixtext_option` statement or the `fixtext_option` statement is omitted
		- A numeric value is right-aligned.
		- The first byte is treated as the sign part: the space character (`0x20`) is output for a positive value and - is output for a negative value.
		- An empty part is filled with `0`.
		Example:
		`Δ00003`
		`-00003`
		Δ : One space character
		• When `format=smallint,type2` is specified in the `fixtext_option` statement
		- A numeric value is right-aligned.
		- The leading digit of the numeric value is treated as the sign part: the space character (`0x20`) is output for a positive value and - is output for a negative value.
		- An empty part is filled with space characters (`0x20`).
		Example:
		`ΔΔΔΔΔ3`
		`ΔΔΔΔ-3`
		Δ : One space character
			6

	`DECIMAL`	• A numeric value is right-aligned in the format *integer.decimal-number*.
		• The first byte is treated as the sign part: the space character (`0x20`) is output for a positive value and - is output for a negative value.
		• An empty part is filled with `0`.
		Example 1:
		`DEC(6,2)`
		`Δ0003.14`
		`-0003.14`
		Example 2:
		`DEC(6,0)`
		`Δ000314.`
		`-000314.`
		Example 3:
		`DEC(6,6)`
		`Δ.000314`
		`-.000314`
		Δ : One space character
			Number of digits + 2

Data type of column		**Output format**	**Output length (bytes)**
	`FLOAT`	• A value is left-aligned in the format *mantissa*`E`*exponent*. • The first byte is treated as the sign part: `+` is output for a positive value and `-` is output for a negative value. • An empty part is filled with space characters (`0x20`). Example 1 `20E10` `+2.000000000000000E+11`Δ Example 2: `-30E222` `-3.000000000000000E+223` Δ : One space character	23
	`SMALLFLT`	• A value is left-aligned in the format *mantissa*`E`*exponent*. • The first byte is treated as the sign part: `+` is output for a positive value and `-` is output for a negative value. • An empty part is filled with space characters (`0x20`). Example: `20E10` `+2.000000000000000E+11`Δ Δ : One space character	23
Character string data	`CHARACTER`	• Left-aligned characters are output up to the defined length. • You can specify whether or not to add enclosing characters with the `enclose` operand in the `fixtext_option` statement. Example 1: `enclose` omitted, defined length is 8 `AIKO`Δ Δ Δ Δ Example 2: `enclose="`, defined length is 8 `"AIKO`Δ Δ Δ Δ`"` Δ : One space character	Defined length*

Data type of column		Output format	Output length (bytes)
	VARCHAR	• Left-aligned characters are output up to the real length. • You can specify whether or not to add enclosing characters with the `enclose` operand in the `fixtext_option` statement. • If the `-W fixtext` option is specified for a variable-length character string that is shorter than the defined length, the data is padded with the padding character up to the defined length. Example 1: `enclose` omitted, defined length is 8, real length is 4 `AIKO`△ △ △ △ Example 2: `enclose="`, defined length is 8, real length is 4 `"AIKO"`△ △ △ △ △ : One space character	Defined length*
Mixed character string data	MCHAR	Same as for `CHARACTER`.	Defined length*
	MVARCHAR	Same as for `VARCHAR`.	Defined length*
National character data	NCHAR	Same as for `CHARACTER`.	Defined length × 2*
	NVARCHAR	Same as for `VARCHAR`.	Defined length × 2*
Date data	DATE	Data is output in the format *yyyy*-*mm*-*dd*. (*yyyy*: year, *mm*: month, *dd*: date). Example: `2004-03-12`	10
Time data	TIME	Data is output in the format *hh*:*mm*:*ss* (*hh*: hour, *mm*: minute, *ss*: second). Example: `12:12:12`	8

Data type of column		Output format	Output length (bytes)
Date interval data	`INTERVAL YEAR TO DAY`	• Data is output in the format `[`Δ`|-]`*yyyymmdd*`.` (Δ: space character, `-`: minus sign, *yyyy*: year, *mm*: month, *dd*: date). • The first byte is treated as the sign part: the space character (`0x20`) is output for a positive value and - is output for a negative value. Example: Δ`00010101.` `-00010101.` Δ: One space character	10
Time interval data	`INTERVAL HOUR TO SECOND`	• Data is output in the format `[`Δ`|-]`*hhmmss*`.` (Δ: space character, `-`: minus sign, *hh*: hour, *mm*: minute, *ss*: second). • The first byte is treated as the sign part: the space character (`0x20`) is output for a positive value and - is output for a negative value. Example: Δ`010101.` `-010101.` Δ: One space character	8
Time stamp data	`TIMESTAMP`	Data is output in the format *yyyy*`-`*mm*`-`*dd*Δ*hh*`:`*mm*`:`*ss*`[.`*nnnnnn*`]` (*yyyy*: year, *mm*: month, *dd*: date, Δ: space character, *hh*: hour, *mm*: minute, *ss*: second, *nnnnnn*: fraction of a second). The fraction of a second is expressed with 0, 2, 4, or 6 digits. When it is expressed with 0 digits, the `.` is not output. Example: `1970-03-12`Δ`12:12:12` Δ: One space character	19 Fractions of a second: 0: +0 2: +3 4: +5 6: +7
Large object data	`BLOB`	No data is output.	0
Binary data	`BINARY`	No data is output.	0
Abstract data type	—	No data is output.	0

Legend:

—: Not applicable

* If you have specified the `enclose` operand in the `fixtext_option` statement, add 2 bytes to the output length.

(b) Rules

1. A maximum of 512 megabytes (536,870,912 bytes) can be output per row (including linefeed characters).
2. If the table contains a column of the abstract data type, you must specify the `unld_func` statement.

8.4 Reloading to a table

8.4.1 Examples

For examples of reloading, see Examples 2 to 5 in Section *8.3.1 Examples*.

8.4.2 Cross-reference by purpose

Required options and control statements depend on the type of reload operation executed by the user.

The options listed in (1) and the control statements are mandatory. The items in (2) through (4) are related to the options and control statements presented in the *Reference* column. For details about the options, see Section 8.9.2.

(1) Required items

Item	Reference			
	Option		Control statement	
Type of `pdrorg` processing	`-k reld`	(1)	—	—
Reorganization of a user-defined table (can be omitted because the default is `-c user`)	`-c user`	(2)	—	—
Name of the table to be reloaded	`-t`	(3)	—	—
File containing the control statements	*control-information-filename*	(19)	—	—
Information about an unload data file	—	—	`unload` statement	8.9.4

— : Not applicable

(2) Information to be specified depending on the attribute of the table being reloaded

Item	Reference location			
	Option		Control statement	
Indexes are defined	`-i`	(6)	`index` statement	8.9.5
	`-o`	(15)	`idxwork` statement	8.9.7
			`sort` statement	8.9.8
There are LOB columns	`-j`	(9)	`lobunld` statement	8.9.9

Item	Reference location			
	Option		Control statement	
There are columns of abstract data type provided by a plug-in (LOB attribute)	`-j`	(9)	`reld_func` statement	8.9.14
There are constraint definitions	—	—	`constraint` statement	8.9.15
Inner replica facility is used	`-q`	(18)	—	—

— : Not applicable

(3) Information to be specified depending on the type of unload data file or LOB data unload file

Item	Reference location			
	Option		Control statement	
Integrating data into a single file (`-g` option specified during unload operation)	`-g`	(8)	—	—
Using EasyMT	`-f`	(12)	`mtguide` statement	8.9.3
			`emtdef` statement	8.9.3
Using HiRDB files	`-f`	(12)	—	—

— : Not applicable

(4) Information to be specified depending on the reloading method

Item	Reference location			
	Option		Control statement	
Reloading only to some of the RDAREAs for a row-partitioned table	`-r`	(4)	—	—
Acquiring or not acquiring database update log	`-l`	(5)	—	—
Specifying synchronization points when reloading a large amount of data	—	—	`option` statement	8.9.16
Monitoring the execution time of `pdrorg`	—	—	`option` statement	8.9.16

Item	Reference location			
	Option		Control statement	
Changing the `pdrorg` executor's user authorization identifier to a value other than the value of environment variable `PDUSER`[1]	`-u`	(10)	—	—
Reloading in batch input/output mode using local buffer instead of global buffer[2]	`-n`	(11)	—	—
Changing a reloading status message output interval to a value other than 100,000 lines	`-m`	(16)	—	—
Monitoring the response time for server-to-server communication	-X	(17)	—	—
Migrating table data to another table	—	—	`tblname` statement	8.9.11
Using the same space character in the entire table data	—	—	`option` statement	8.9.16
Changing the percentage of free space in table and index during reloading[3]	—	—	`option` statement	8.9.16
Obtaining tuning information during the execution of `pdrorg`	—	—	`report` statement	8.9.17

— : Not applicable

[1] If omitted, the system assumes the value of the `PDUSER` environment variable. If the `PDUSER` environment variable has not been specified, the system assumes the user name in the login window.

[2] By specifying the number of batch input/output pages, you can reduce the number of I/O operations because data is input/output in units of specified pages in batch mode.

[3] If a space shortage occurs in an RDAREA during execution of `pdrorg`, you can complete the processing without having to expand the RDAREA temporarily (except for a table or index whose percentage of free space is 0).

8.4.3 Whether or not reloading is permitted when the table definitions of unload table and reload table do not match

`pdrorg` outputs the definition information for an unloaded table to the unload data file.

During reloading, `pdrorg` compares the table definition information in the unload data file with that for the table to be reloaded to determine whether reloading is possible.

If the table definitions are the same, the unloaded table can be reloaded. However, whether or not reloading to a different table, transfer of table data to another system, or reloading to a table with a different table definition is possible depends on conditions.

(1) Options to be specified during reloading

If you are reloading table data to a different table, transferring table data to another system, or reloading table data to a table whose table storage conditions are different from those of the unloaded table, you must specify the following options:

- During unloading and reloading, specify the `-g` option.
- If the table contains a BLOB column or a column of an abstract data type with the `BLOB` attribute, specify both the `-j` option and the `-g` option.

(2) Comparing the storage conditions between the unloaded table and the reload table

If the unloaded table and the reload table employ the same partitioning method, the utility checks the items listed in Table 8-4 to determine whether or not their storage conditions are different. If their storage conditions are different, whether or not reloading is possible is determined by the criteria shown in Table 8-5 (this is applicable when reloading is to be performed in units of tables or schemas; reloading in units of RDAREAs is not supported at all).

Table 8-4: Storage condition check items when the unloaded table and reload table employ the same partitioning method

Partitioning method	Items checked	
Non-partitioning	None	
Hash partitioning	Hash function	Function name
	Hash type	FIX or flexible
	Column structure of partitioning key	Defined order
	Storage RDAREAs	Number of storage RDAREAs, RDAREA IDs, RDAREA names, and server names

Partitioning method	Items checked	
Key range partitioning	Column structure of partitioning key	Defined order and column IDs
	Start column of partitioning key	Column ID, data type, and defined length
	Number of partitioning conditions	
	Partitioning condition	Storage RDAREA IDs and condition values
Matrix partitioning	Column structure of partitioning key	Defined order
	Number of partitioning conditions	
	Number of partitioning condition values	
	Partitioning key conditions	Number of partitions in key, dimension number, partitioning column ID, data type, defined length, length of condition value, and condition value
	Hash function	Function name
	Hash type	FIX or flexible
	Storage RDAREAs	Number of storage RDAREAs, RDAREA IDs, RDAREA names, and server names

Table 8-5: Whether or not reloading is possible when storage conditions are different

-g[1]	**HiRDB conf**	**Partitioning conditions of unloaded table**		**Partitioning conditions of table to be reloaded**					
				Key range ptn	**Hash partitioning**		**Matrix partitioning**		**Not partitioned**
					FIX	**Flex**	**Key range and key range**	**Key range and hash**	
Specified	HiRDB/ Single Server	Key range partitioning		Y	Y	Y	Y	Y	Y
		Hash ptn	FIX	Y	Y	Y	Y	Y	Y
			Flex	Y	Y	Y	Y	Y	Y
		Matrix ptn	Key range and key range	Y	Y	Y	Y	Y	Y
			Key range and hash	Y	Y	Y	Y	Y	Y
		Not partitioned		Y	Y	Y	Y	Y	Y
	HiRDB/ Parallel Server	Key range partitioning		Y	Y	Y	Y	Y	Y
		Hash ptn	FIX	Y	Y	Y	Y	Y	Y
			Flex	Y	Y	Y	Y	Y	Y
		Matrix ptn	Key range and key range	Y	Y	Y	Y	Y	Y
			Key range and hash	Y	Y	Y	Y	Y	Y
		Not partitioned		Y	Y	Y	Y	Y	Y

-g[1]	HiRDB conf	Partitioning conditions of unloaded table		Partitioning conditions of table to be reloaded					
				Key range ptn	Hash partitioning		Matrix partitioning		Not partitioned
					FIX	Flex	Key range and key range	Key range and hash	
Omitted	HiRDB/ Single Server	Key range partitioning		Y	Y	Y	Y	Y	Y
		Hash ptn	FIX	N	Y	Y	Y	Y	Y
			Flex	N	Y	Y	Y	Y	Y
		Matrix ptn	Key range and key range	Y	Y	Y	Y	Y	Y
			Key range and hash	Y	Y	Y	Y	Y	Y
		Not partitioned		Y	Y	Y	Y	Y	Y
	HiRDB/ Parallel Server	Key range partitioning		Y[6]	N	Y[2, 3]	N	N	Y[5]
		Hash ptn	FIX	N	N	Y[2, 4]	N	N	Y[5]
			Flex	N	N	Y[7]	N	N	Y[5]
		Matrix ptn	Key range and key range	N	N	Y[2, 3]	Y[6]	N	Y[5]
			Key range and hash	N	N	Y[2, 3]	N	N	Y[5]
		Not partitioned		N	N	Y[2, 3]	N	N	Y[2]

Legend:

Y: Can be reloaded according to the storage conditions of the table to be reloaded.

N: Cannot be reloaded.

conf: configuration

ptn: partitioning

Flex: flexible

[1] Indicates whether or not the `-g` option is specified during unloading or reloading. Reloading from a partitioned table to a non-partitioned table when the `-g` option is omitted means reloading from a partitioned table that is not partitioned among multiple servers to a non-partitioned table. If the table is partitioned among multiple servers, specify the `-g` option.

[2] For a table with a BLOB column or a column of an abstract data type with the `BLOB` attribute, reloading is supported only when the `-j` option is specified during unloading and reloading.

[3] `pdrorg` stores data in a wrap-around manner without using the hash function.

[4] Data is stored in the same RDAREA as during unloading. Data cannot be reloaded if the corresponding RDAREA is not found during reloading.

[5] Reloading is not supported for a table containing a BLOB column or a column of an abstract data type with the `BLOB` attribute

[6] Whether or not reloading is supported depends on the change made to the table storage conditions, as described below:

Change to the table storage conditions		**Whether or not reloading is supported**	**Remarks**
Partitioning conditions changed	Partitioning conditions added	N	Terminates with an error.
	Partitioning conditions deleted		
	Partitioning key value changed		

Change to the table storage conditions			Whether or not reloading is supported	Remarks
Partitioning conditions not changed	Storage RDAREA changed	RDAREA name changed	Y	If the conditions of [2] and [5] are satisfied, the utility stores data in the RDAREA corresponding to the storage RDAREA used during unloading.
		Change made to the server containing the RDAREA	N	Terminates with an error.
	Storage RDAREA not changed	Partitioning key component column changed (such as column name and specification order)	N	Terminates with an error.

[7] Whether or not reloading is supported depends on the change made to the table storage conditions, as described below:

Change to the table storage conditions		Whether or not reloading is supported	Remarks
Storage RDAREA changed	RDAREA added	Y	If the conditions of [4] and [5] are satisfied, the utility stores data in the RDAREA used during unloading. Data is not stored in the added RDAREA.
	RDAREA deleted	N	Terminates with an error.
	RDAREA renamed		
	Change made to the server containing the RDAREA		
Storage RDAREA not changed	Hash function changed	Y	If the conditions of [4] and [5] are satisfied, the utility stores data in the RDAREA used during unloading.
	Partitioning key component column changed (such as column name and specification order)		

(3) Limitations on reloading

Reloading is prohibited in the following cases:

1. For a table containing a LOB column, the `-j` option is omitted.
2. Reloading from a FIX table to a non-FIX table, or vice versa.
3. There is a mismatch between the unloaded table and the table to be reloaded in terms of the number of table columns, the order of the column definitions, the data types of a column, the column names, or the column attributes (such as `NOT NULL`).
4. The unload table contains a column with the `NOT NULL` attribute and the reload table contains partitioning key structure columns. (If a column with a `NOT NULL` attribute is changed to a partitioning key structure column, the corresponding table must be unloaded with the `-W` option specified. Then the data can be loaded with the database load utility.)
5. A new unique key index or primary key index is defined for a data column with duplicated values.
6. When the number of elements for each repetition column in the table to be reloaded is less than the unloaded table.
7. The data type of a column for which an abstract data type has been defined has been changed.

8.4.4 Handling of a reloading error during table reorganization

If reloading results in an error during table reorganization, eliminate the cause of the error and then re-execute table reorganization (`-k rorg`).

(1) Processing resulting in an error during table reorganization and the restart location for re-execution

If table reorganization results in an error, the table is placed in *under-reorganization* status. If table reorganization is re-executed while the table's status is under-reorganization, processing is restarted with the process that resulted in the error. To determine whether or not the corresponding RDAREA was being reorganized, execute `pddbst`'s logical analysis in units of RDAREAs or analysis in units of tables and then check `Status` in the analysis results.

(a) When reorganizing a LOB column structure base table only or together with a LOB column (-j specified)

In the event of an error during table reorganization, Table 8-6 describes the location where table reorganization is restarted. Reorganization of a LOB column structure base table only or together with a LOB column is based on the assumption that the `-j` option is specified and a single unload data file is used.

Specification of options for execution of `pdrorg`:

```
pdrorg -k rorg [-j]
```

Specification of a control information file for execution of `pdrorg`:

`unload` statement

Table 8-6: Restart location when table reorganization is re-executed due to an error (when only LOB column structure base table or both LOB column structure base table and LOB column are reorganized (-j specified))

No.	**Processing during table reorganization**		**Output message immediately before error***		**Table status**			**Restart location**
					LOB column structure base table	**LOB column**	**LOB attribute**	
1	Unloading	Starting unloading	`712`, `732`		—	—	—	No. 1
2		Completing unloading	`714`, `734`	`rc=8`	—	—	—	No. 1
				`rc=0`	N and R	N and R	N and R	For falsification prevented table: No. 3
					R	R	R	For table other than falsification prevented table: No. 1 (in units of tables or RDAREAs) No. 3 (in units of schemas)

No.	Processing during table reorganization		Output message immediately before error*		Table status			Restart location
					LOB column structure base table	LOB column	LOB attribute	
3	Reloading	Starting data deletion	Not outpu t		N and R	N and R	N and R	For falsification prevented table: No. 3
					R	R	R	For table other than falsification prevented table: No. 1 (in units of tables or RDAREAs) No. 3 (in units of schemas)
4		Completing data deletion	`721`		N and R	N and R	N and R	No. 5
5		Starting reloading	`712, 732`		N and R	N and R	N and R	No. 5
6		Completing reloading	`714, 734`	`rc=8`	N and R	N and R	N and R	No. 5
				`rc=0`	R	R	R	When index has been defined and `-i c` is specified: No. 7 Other than the above: No. 1

No.	Processing during table reorganization		Output message immediately before error*	Table status			Restart location
				LOB column structure base table	LOB column	LOB attribute	
7	Batch index creation	Starting index unloading (during restart only)	`725`	R	R	R	No. 7
8		Starting batch index creation	`715`	R	R	R	No. 7
9		Completing batch index creation	`716`	R	R	R	When there is an index that has not been created: No. 7
				—	—	—	Other than the above: No. 1

Legend:

N: Reload-not-completed data status

R: Reorganizing

—: Normal status (before or after reorganization)

* Messages are abbreviated. For example, `712` means `KFPL00712-I` and `732` means `KFPL00732-I`.

(b) When reorganizing a LOB column structure base table together with a LOB column (lobunld statement specified)

In the event of an error during table reorganization, Table 8-7 describes the location where table reorganization is restarted. Reorganization of a LOB column structure base table only or together with a LOB column is based on the assumption that the `lobunld` statement is specified and an unload data file and LOB data unload data file are used.

Specification of options for execution of `pdrorg`:

```
pdrorg -k rorg
```

Specification of a control information file for execution of `pdrorg`:

`unload` and `lobunld` statements

Table 8-7: Restart location when table reorganization is re-executed due to an error (when only LOB column structure base table or both LOB column structure base table and LOB column are reorganized (lobunld statement specified))

No.	Processing during table reorganization		Output message immediately before error*		Table status			Restart location
					LOB column structure base table	LOB column	LOB attribute	
1	Unloading LOB column structure base table	Starting unloading	`712, 732`		—	—	—	No. 1
2		Completing unloading	`714, 734`	`rc=8`	—	—	—	No. 1
				`rc=0`	N and R	N and R	N and R	For falsification prevented table: No. 3
					R	R	R	For table other than falsification prevented table: No. 1
3	Reloading to LOB column structure base table	Starting data deletion	Not output		N and R	N and R	N and R	For falsification prevented table: No. 3
					R	R	R	For table other than falsification prevented table: No. 1
4		Completing data deletion	`721`		N and R	N and R	N and R	No. 5
5		Starting reloading	`712, 732`		N and R	N and R	N and R	No. 5

No.	Processing during table reorganization		Output message immediately before error*		Table status			Restart location
					LOB column structure base table	LOB column	LOB attribute	
6		Completing reloading	`714`, `734`	`rc=8`	N and R	N and R	N and R	No. 5
				`rc=0`	R	R	R	When index has been defined and `-i c` is specified: No. 7 Other than the above: No. 1
7	Batch index creation	Starting index unloading (during restart only)	`725`		R	R	R	No. 7
8		Starting batch index creation	`715`		R	R	R	No. 7
9		Completing batch index creation	`716`		R	R	R	When there is an index that has not been created: No. 7 Other than the above: No. 10

No.	Processing during table reorganization		Output message immediately before error*		Table status			Restart location
					LOB column structure base table	LOB column	LOB attribute	
10	Unloading LOB column	Starting unloading	`712,732`		R	R	R	No. 10
11		Completing unloading	`714, 734`	`rc=8`	R	R	R	No. 10
				`rc=0`	R	N and R	R	For falsification prevented table: No. 12
					R	R	R	For table other than falsification prevented table: No. 10
12	Reloading LOB column	Starting data deletion	Not output		R	N and R	R	For falsification prevented table: No. 12
					R	R	R	For table other than falsification prevented table: No. 10
13		Completing data deletion	`721`		R	N and R	R	No. 14
14		Starting reloading	`712,732`		R	N and R	R	No. 14
15		Completing reloading	`714, 734`	`rc=8`	R	N and R	R	No. 14
				`rc=0`	—	—	—	No. 1

Legend:

N: Reload-not-completed data status

R: Reorganizing

—: Normal status (before or after reorganization)

* Messages are abbreviated. For example, `712` means `KFPL00712-I` and `732` means `KFPL00732-I`.

(c) When reorganizing a LOB column only

In the event of an error during table reorganization, Table 8-8 describes the location where table reorganization is restarted.

Specification of options for execution of `pdrorg`:

```
pdrorg -k rorg
```

Specification of a control information file for execution of `pdrorg`:

`lobunld` statement

Table 8-8: Restart location when table reorganization is re-executed due to an error (when LOB column only is reorganized)

No.	Processing during table reorganization		Output message immediately before error*		Table status			Restart location
					LOB column structure base table	LOB column	LOB attribute	
1	Unloading LOB column	Starting unloading	`712`, `732`		—	—	—	No. 1
2		Completing unloading	`714`, `734`	`rc=8`	—	—	—	No. 1
				`rc=0`	—	N and R	—	For falsification prevented table: No. 3
					—	R	—	For table other than falsification prevented table: No. 1

No.	Processing during table reorganization		Output message immediately before error*		Table status			Restart location
					LOB column structure base table	LOB column	LOB attribute	
3	Reloading LOB column	Starting data deletion	Not outpu t		—	N and R	—	For falsification prevented table: No. 3
					—	R	—	For table other than falsification prevented table: No. 1
4		Completing data deletion	`721`		—	N and R	—	No. 5
5		Starting reloading	`712`, `732`		—	N and R	—	No. 5
6		Completing reloading	`714`, `734`	`rc=8`	—	N and R	—	No. 5
				`rc=0`	—	—	—	No. 1

Legend:

N: Reload-not-completed data status

R: Reorganizing

—: Normal status (before or after reorganization)

* Messages are abbreviated. For example, `712` means `KFPL00712-I` and `732` means `KFPL00732-I`.

(d) When reorganizing in units of schemas

In the event of an error during table reorganization, Table 8-9 describes the location where table reorganization is restarted.

Specification of options for execution of `pdrorg`:

```
pdrorg -k rorg -t all
```

Table 8-9: Restart location when table reorganization is re-executed due to an error (reorganization in units of schemas)

No.	Processing during table reorganization	Timing	Restart location
1	Unloading	When an error occurred during unloading	No. 1
2	Reloading	When an error occurred after the `KFPL00722-I` message was issued	No. 2

(e) When reorganizing for each server (-g omitted) (applicable to HiRDB/Parallel Server only)

Specification of options for execution of `pdrorg`:

```
pdrorg -k rorg
```

Specification of a control information file for execution of `pdrorg`:

`unload` statement only, `lobunld` statement only, or both `unload` and `lobunld` statements

For a HiRDB/Parallel Server, if reloading results in an error during reorganization of a table that is split among multiple servers, reorganization is restarted at the location indicated in (a) through (c) for each server where the `unload` and `lobunld` statements are specified in the control information file.

(2) Whether or not commands and SQL statements can be executed on a table that is in reload-not-completed data status

If reloading results in an error during table reorganization or reorganization is incomplete for a reason such as an error during reorganization of a falsification prevented table, the target RDAREA is placed in reload-not-completed data status. In such a case, execution of some commands and SQL statements is restricted.

Table 8-10 describes whether or not utilities can be executed on a table that is in reload-not-completed data status, Table 8-11 describes whether or not operation commands can be executed on such a table, and Table 8-12 describes whether or not SQL statements can be executed on such a table.

Table 8-10: Whether or not utilities can be executed on a table in reload-not-completed data status

Utility name	**Function**			**Table status**							
				B: —	**B: RN**	**B: —**	**B: —**	**B: RN**	**B: RN**	**B: —**	**B: N**
				A: —	**A: —**	**A: RN**	**A: —**	**A: RN**	**A: —**	**A: RN**	**A: N**
				L: —	**L: —**	**L: —**	**L: RN**	**L: —**	**L: RN**	**L: RN**	**L: N**
`pdload`	Data loading on LOB column structure base table		`-d` specified	Y	Y	Y	Y	Y	Y	Y	Y
			`-d` omitted	Y	N	N	N	N	N	N	N
	Data loading on LOB column			Y	N	N	N	N	N	N	N
	Concurrent data loading on LOB column structure base table and LOB column		`-d` specified	Y	Y	Y	Y	Y	Y	Y	Y
			`-d` omitted	Y	N	N	N	N	N	N	N
`pdrorg`	Table reorganization (`-k rorg`)	`-j` specified	`unload` statement	Y	C	C	C	C	C	C	C
		`-j` omitted	`unload` statement	Y	C	C	C	C	C	C	C
			`lobunld` statement	Y	C	C	C	C	C	C	C
			`unload` statement and `lobunld` statement	Y	C	C	C	C	C	C	C
	Unloading table (`-k unld`)			Y	Y	Y	Y	Y	Y	Y	Y
	Table reloading (`-k reld`)	`-j` specified	`unload` statement	Y	Y	Y	Y	Y	Y	Y	Y
		`-j` omitted	`unload` statement	Y	Y	Y	Y	Y	Y	Y	Y
			`lobunld` statement	Y	Y	Y	Y	Y	Y	Y	Y

Utility name	**Function**			**Table status**							
				B: —	**B: RN**	**B: —**	**B: —**	**B: RN**	**B: RN**	**B: —**	**B: N**
				A: —	**A: —**	**A: RN**	**A: —**	**A: RN**	**A: —**	**A: RN**	**A: N**
				L: —	**L: —**	**L: —**	**L: RN**	**L: —**	**L: RN**	**L: RN**	**L: N**
			`unload` statement and `lobunld` statement	Y	Y	Y	Y	Y	Y	Y	Y
	Batch index creation (`-k ixmk`)		B-TREE index	Y	N	N	N	N	N	N	N
			Plug-in index	Y	N	N	N	N	N	N	N
	Index re-creation (`-k ixrc`)		B-TREE index	Y	N	N	N	N	N	N	N
			Plug-in index	Y	N	N	N	N	N	N	N
	Index reorganization (`-k ixor`)			Y	N	N	N	N	N	N	N
`pdrbal`				Y	N	N	N	N	N	N	N
Other utility				Y	Y	Y	Y	Y	Y	Y	Y

Legend:

B: RDAREA storing LOB column structure base table

A: RDAREA containing LOB attributes

L: RDAREA containing LOB columns

RN: Reload-not-completed data status

— : Normal status (before or after reorganization)

Y: Can be executed.

C: Can be re-executed if no change is made to the options and control information files. If changes are made to the options or control information files, re-re-execution results in an error.

N: Cannot be executed.

Table 8-11: Whether or not operation commands can be executed on a table in reload-not-completed data status

Operation command name	**Condition**		**Table status**							
			B: —	**B: RN**	**B: —**	**B: —**	**B: RN**	**B: RN**	**B: —**	**B: RN**
			A: —	**A: —**	**A: RN**	**A: —**	**A: RN**	**A: —**	**A: RN**	**A: RN**
			L: —	**L: —**	**L: —**	**L: RN**	**L: —**	**L: RN**	**L: RN**	**L: RN**
`pdrels`	RDAREA containing falsification prevented table	B	Y	N	Y	Y	N	N	Y	N
		A	Y	Y	N	Y	N	Y	N	N
		L	Y	Y	Y	N	Y	N	N	N
	Other RDAREA		Y	Y	Y	Y	Y	Y	Y	Y
Other operation command			Y	Y	Y	Y	Y	Y	Y	Y

Legend:

B: RDAREA containing LOB column structure base table

A: RDAREA containing LOB attributes

L: RDAREA containing LOB columns

RN: Reload-not-completed data status

—: Normal status (before or after reorganization)

Y: Can be executed.

N: Cannot be executed.

Table 8-12: Whether or not SQL statements can be executed on a table in reload-not-completed data status

SQL	**Table status**							
	B: —	**B: RN**	**B: —**	**B: —**	**B: RN**	**B: RN**	**B: —**	**B: RN**
	A: —	**A: —**	**A: RN**	**A: —**	**A: RN**	**A: —**	**A: RN**	**A: RN**
	L: —	**L: —**	**L: —**	**L:RN**	**L: —**	**L:RN**	**L:RN**	**L:RN**
`SELECT` statement	Y	Y	C	C	C	C	C	C
`UPDATE` statement	Y	Y	C	C	C	C	C	C

SQL	Table status							
	B: —	**B: RN**	**B: —**	**B: —**	**B: RN**	**B: RN**	**B: —**	**B: RN**
	A: —	**A: —**	**A: RN**	**A: —**	**A: RN**	**A: —**	**A: RN**	**A: RN**
	L: —	**L: —**	**L: —**	**L:RN**	**L: —**	**L:RN**	**L:RN**	**L:RN**
`INSERT` statement	Y	Y	C	C	C	C	C	C
`DELETE` statement	Y	Y	C	C	C	C	C	C
Other SQL statement	Y	Y	Y	Y	Y	Y	Y	Y

Legend:

B: RDAREA storing LOB column structure base table

A: RDAREA storing LOB attributes

L: RDAREA containing LOB columns

RN: Reload-not-completed data status

—: Normal status (before or after reorganization)

Y: Can be executed.

C: If a LOB column or a column of an abstract data type with the LOB attribute is to be manipulated, the SQL statement cannot be executed if the RDAREA storing the target LOB column or LOB attribute is in reload-not-completed data status.

(3) Reload-not-completed data status release timing

The reload-not-completed data status is released at the following times:

- When the `PURGE TABLE` statement is executed
- When `pdmod` is used to re-initialize the RDAREA
- When table reorganization (`-k rorg`) is re-executed using the `option` statement with `tblstatus=clear` specified, and `pdrorg` terminates normally
- When table reorganization (`-k rorg`) is re-executed and `pdrorg` terminates normally
- When table reloading (`-k reld`) is executed and `pdrorg` terminates normally
- When `pdload` (`-d` specified) is executed

(4) Notes

1. Table reorganization (`-k rorg`) is not restarted in the following cases; execute

table reloading (`-k reld`) or recover the table from its backup:

- In the no-log mode (`-l n` specified)
- When a data dictionary table was reorganized
- When changes were made to the options and control information file (except for changes to the `-b` option or to the `option` statement's `tblstatus` operand)
- Before re-execution, the following processing was executed on the target table (in this case, start from unload processing instead of restarting the processing):

  - Execution of `pdload` with `-d` specified

  - Recovery of RDAREA from a backup created prior to the table reorganization

  - Reloading on the table (`-k reld`)

  - Re-initialization of RDAREA by `pdmod`

  - Execution of the `PURGE TABLE` statement

2. Do not execute `pdrorg` from any other user until the current table reorganization is completed (do not concurrently perform re-execution of table reorganization, normal table reorganization, or reloading on the table).
3. If indexes are defined for a table and table reorganization is re-executed in the index update mode (`-i s`), the index information file may remain. In such a case, delete the index information file after table reorganization has been completed.
4. If you are re-executing table reorganization in units of schemas, shut down the RDAREA storing the table in the schema with the `pdhold` command (to protect UAPs from accessing the table being reorganized and being placed in wait status and to protect the tables from being updated by UAPs). Additionally, do not execute `CREATE TABLE` or `DROP TABLE` on the corresponding schema until table reorganization for that schema is completed.
5. Table reorganization cannot be restarted if the `-r` option is specified, a non-partitioning key index is specified in the `-b` option, and the restart location falls on the unloading of LOB column structure base table or unloading of LOB column. In such a case, delete the `-b` option and then re-execute or execute table reloading (`-k reld`).

8.5 Reorganizing data dictionary tables

8.5.1 Example

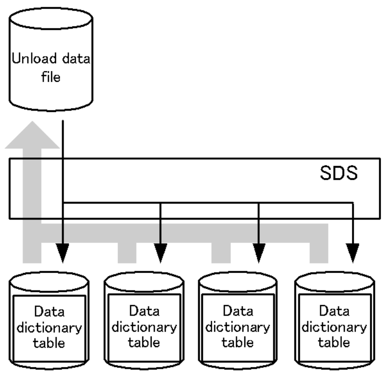
This section presents an example of using the database reorganization utility (reorganizing a data dictionary table).

Example 1

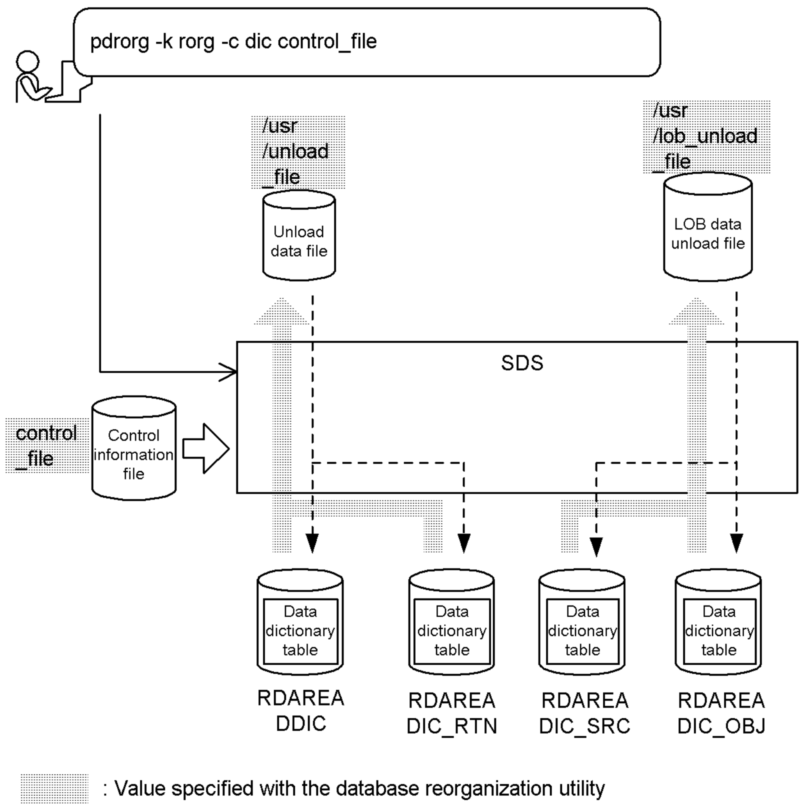
This example reorganizes data dictionary tables in units of tables. The example assumes that the data dictionary tables are stored in the RDAREAs shown as follows. When reorganizing data dictionary tables, be sure to shut down the RDAREAs or LOB RDAREAs that contain the data dictionary tables being reorganized.

- Data dictionary RDAREAs (`DDIC` and `DIC_RTN`)
- Data dictionary LOB RDAREAs (`RTN_SRC` and `RTN_OBJ`)

Overview



Relationship between input/output files and RDAREAs



Explanation of the command

This example reorganizes the data dictionary tables in units of tables.

`-k rorg`: Specification for unloading

`-c dic`: Specification for reorganizing data dictionary tables

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
unload /usr/unload_file                   1
lobunld /usr/lob_unload_file              2
```

Explanation:

1. Specifies the unload data file:

   `/usr/unload_file`: Name of the unload data file

2. Specifies the LOB data unload file:

   `/usr/lob_unload_file`: Name of the LOB data unload file

8.5.2 Cross-reference by purpose

Required options and control statements depend on the type of reorganization.

The options listed in (1) and the control statements are mandatory. The items in (2) and (3) are related to the options and control statements presented in the *Reference* column. For details about the options, see Section 8.9.2.

(1) Required items

Item	Reference			
	Option		Control statement	
Type of `pdrorg` processing	`-k rorg`	(1)	—	—
Reorganization of data dictionary table	`-c dic`	(2)	—	—
File containing the control statements	*control-information-filename*	(19)	—	—
Information about an unload data file	—	—	`unload` statement	8.9.4

— : Not applicable

(2) Information to be specified depending on the type of unload data file

Item	Reference location			
	Option		Control statement	
Using EasyMT	`-f`	(12)	`mtguide` statement	8.9.3
			`emtdef` statement	8.9.3
Using HiRDB files	`-f`	(12)	—	—

— : Not applicable

(3) Information to be specified depending on the reorganization method

Item	Reference location			
	Option		Control statement	
Reorganizing only a specified data dictionary table	`-t`	(3)	—	—
Acquiring or not acquiring database update log	`-l`	(5)	—	—
Monitoring the execution time of `pdrorg`	—	—	`option` statement	8.9.16
Changing the `pdrorg` executor's user authorization identifier to a value other than the value of environment variable `PDUSER`*	`-u`	(10)	—	—
Changing a reorganization status message output interval to a value other than 100,000 lines	`-m`	(16)	—	—
Monitoring the response time for server-to-server communication	-X	(17)	—	—

— : Not applicable

* If omitted, the system assumes the value of the `PDUSER` environment variable. If the `PDUSER` environment variable has not been specified, the system assumes the user name in the login window.

8.6 Creating indexes in batch mode

8.6.1 Examples

This section presents examples of using the database reorganization utility (creating indexes in batch mode), listed as follows:

Example	Description	Classification
1	Creating an index in batch mode • Creating an index in batch mode on the basis of the index information file output in table reorganization example 2	S
2	Creating a plug-in index in delayed batch mode • Creating a plug-in index defined for a table with a abstract data type (`SGMLTEXT`) in delayed batch mode	
3	Creating indexes in batch mode • Using the index information files output during the execution of the database load utility (index information output mode)	P

S: HiRDB/Single Server

P: HiRDB/Parallel Server

(1) Creating an index in batch mode

Example 1

This example creates an index (`INDEX2`) in batch mode on the basis of the index information that was output in table reorganization example 2. The name of the index information file is `/usr/index_inf2`. Index information in `PDBUSER02` has already been obtained in `/usr/index_inf4`.

The example assumes that the following table (`TABLE1`) and indexes have been defined:

- Table definition:

```
CREATE TABLE TABLE1(C1 INT NOT NULL,C2 CHAR(8),C3 INT)
        IN ((PDBUSER01) C1 > 10,(PDBUSER02))
```

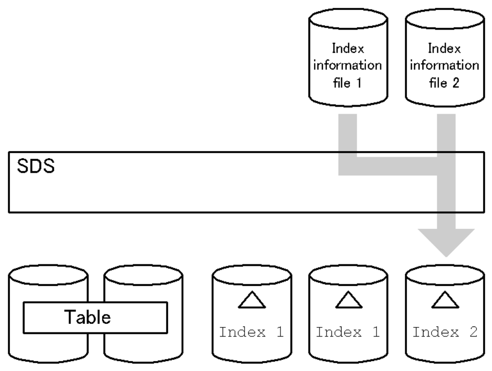
- Index definition (partitioning key index):

```
CREATE INDEX INDEX1 ON TABLE1(C1)
        IN ((PDBUSER03),(PDBUSER04))
```

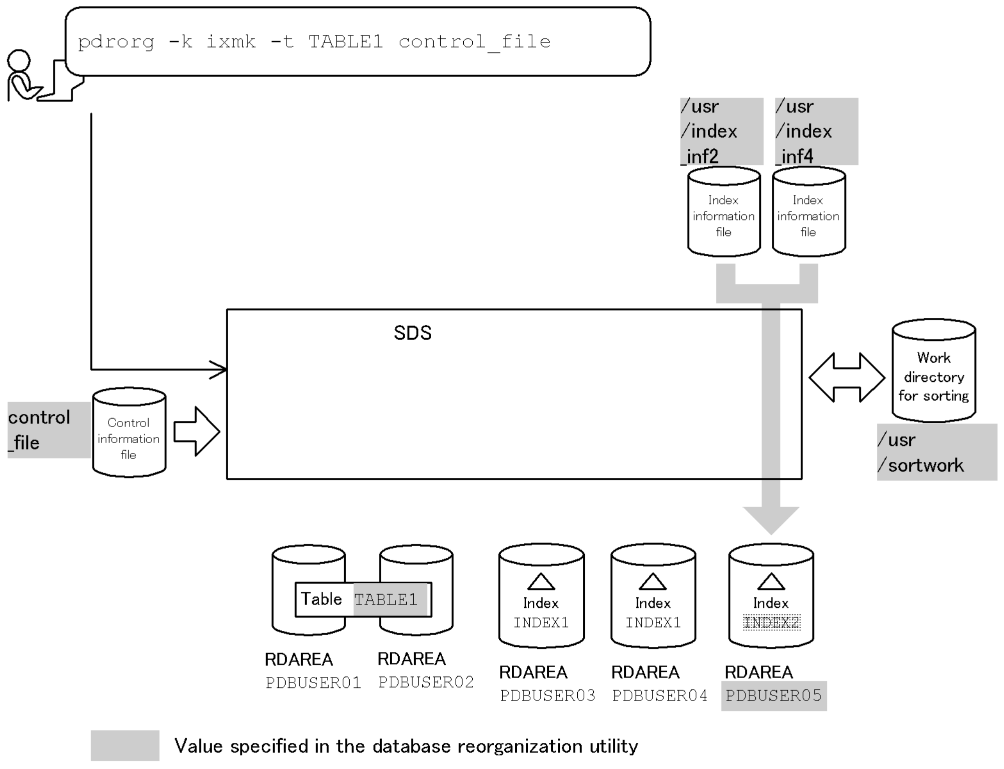
- Index definition (non-partitioning key index):

```
CREATE INDEX INDEX2 ON TABLE1(C2,C1) IN (PDBUSER05)
```

Overview



Relationship between input/output files and RDAREAs



Explanation of the command

This example creates an index (`INDEX2`) in batch mode on the basis of the index information.

`-k ixmk`: Specification for batch index creation

`-t TABLE1`: Name of the table subject to batch index creation

`-o`: Specification for deleting unneeded index information files after index creation is completed

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
index INDEX2 PDBUSER05 /usr/index_inf2              1
index INDEX2 PDBUSER05 /usr/index_inf4              1
sort /usr/sortwork,8192                             2
```

Explanation:

1. Specifies the index information files containing the index information:

   `INDEX2`: Index identifier

   `PDBUSER05`: Name of the index storage RDAREA

   `/usr/index_inf2, /usr/index_inf4`: Names of the index information files

2. Specifies the work directory for sorting:

   `/usr/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)

(2) Creating a plug-in index in delayed batch mode

Example 2

This example creates a plug-in index (`INDEX1`) in delayed batch mode that is defined for a table (`TABLE1`) with columns of abstract data type (`SGMLTEXT`). The abstract data type is provided by the HiRDB Text Search Plug-in.

The example assumes that the following table and index have been defined:

- Table definition:

```
CREATE TABLE TABLE1(C1 INT,C2 SGMLTEXT
          ALLOCATE (SGMLTEXT IN LOBUSER01)
          PLUGIN'<DTD>sgml.dtd</DTD>' ) IN PDBUSER01
```

- Plug-in index definition:

```
CREATE INDEX INDEX1 USING TYPE NGRAM ON TABLE1(C2)
```

```
            IN LOBUSER02
```

The index information file was created by executing an updating UAP (`INSERT` or `UPDATE` statement) with the following definition information:

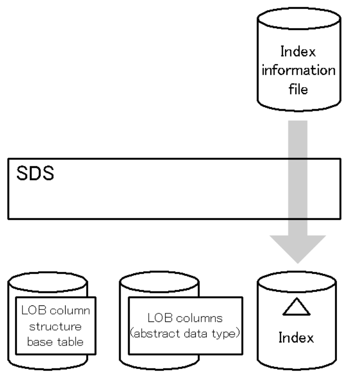
- Server definition:

```
set pd_plugin_ixmk_dir="/hd001/ixdir"
```

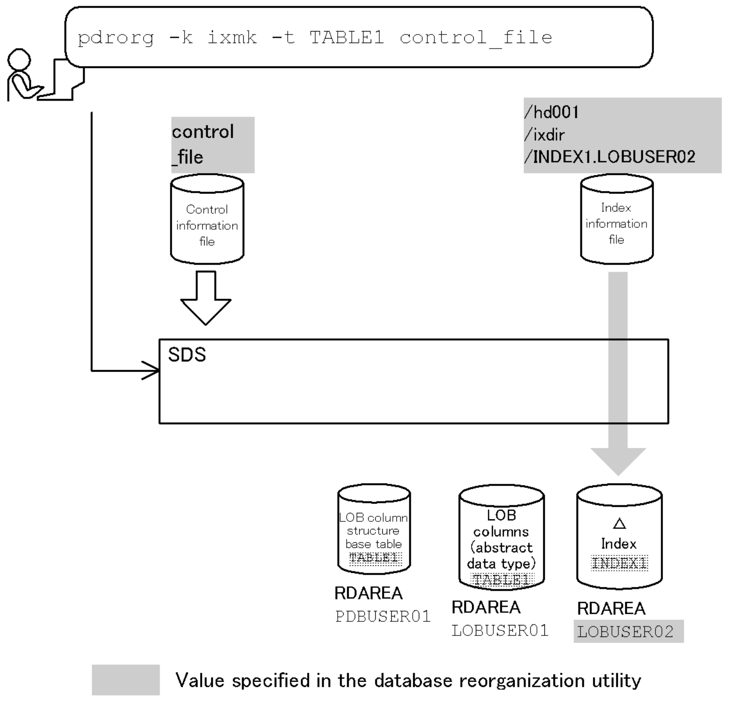
- Client environment variable:

```
PDPLGIXMK YES
```

Overview


Index information file
SDS
LOB column structure base table
LOB columns (abstract data type)
Index

Relationship between input/output files and RDAREAs


pdrorg -k ixmk -t TABLE1 control_file
control_file
Control information file
/hd001/ixdir/INDEX1.LOBUSER02
Index information file
SDS
LOB column structure base table TABLE1
RDAREA PDBUSER01
LOB columns (abstract data type) TABLE1
RDAREA LOBUSER01
△ Index INDEX1
RDAREA LOBUSER02
Value specified in the database reorganization utility

Explanation of the command

This example creates a plug-in index (`INDEX1`) defined for the table (`TABLE1`) in delayed batch mode.

`-k ixmk`: Specification for batch index creation

`-t TABLE1`: Name of the table subject to batch index creation

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
index INDEX1 LOBUSER02 /hd001/ixdir/INDEX1.LOBUSER02              1
```

Explanation:

1. Specifies the index information file containing the information subject to delayed batch index creation

   `INDEX1`: Identifier of the plug-in index subject to delayed batch index creation

`LOBUSER02`: Name of the index storage RDAREA for the plug-in index subject to delayed batch index creation

`/hd001/ixdir/INDEX1.LOBUSER02`: Name of the index information file

(3) Creating indexes in batch mode

Example 3

This example creates indexes in batch mode on the basis of the index information files. It uses four index information files that were output during the execution of the database load utility (to load data in index information output mode).

The example assumes that the following table and indexes have been defined:

- Table definition:

```
CREATE TABLE TABLE1(C1 INT NOT NULL,C2 CHAR(8),C3 INT)
               IN ((PDBUSER01) C1 > 10,(PDBUSER02))
```

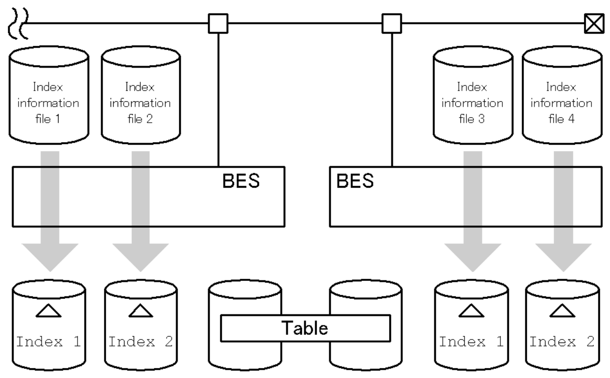
- Index definition (partitioning key index):

```
CREATE INDEX INDEX1 ON TABLE1(C1)
               IN ((PDBUSER03),(PDBUSER05))
```

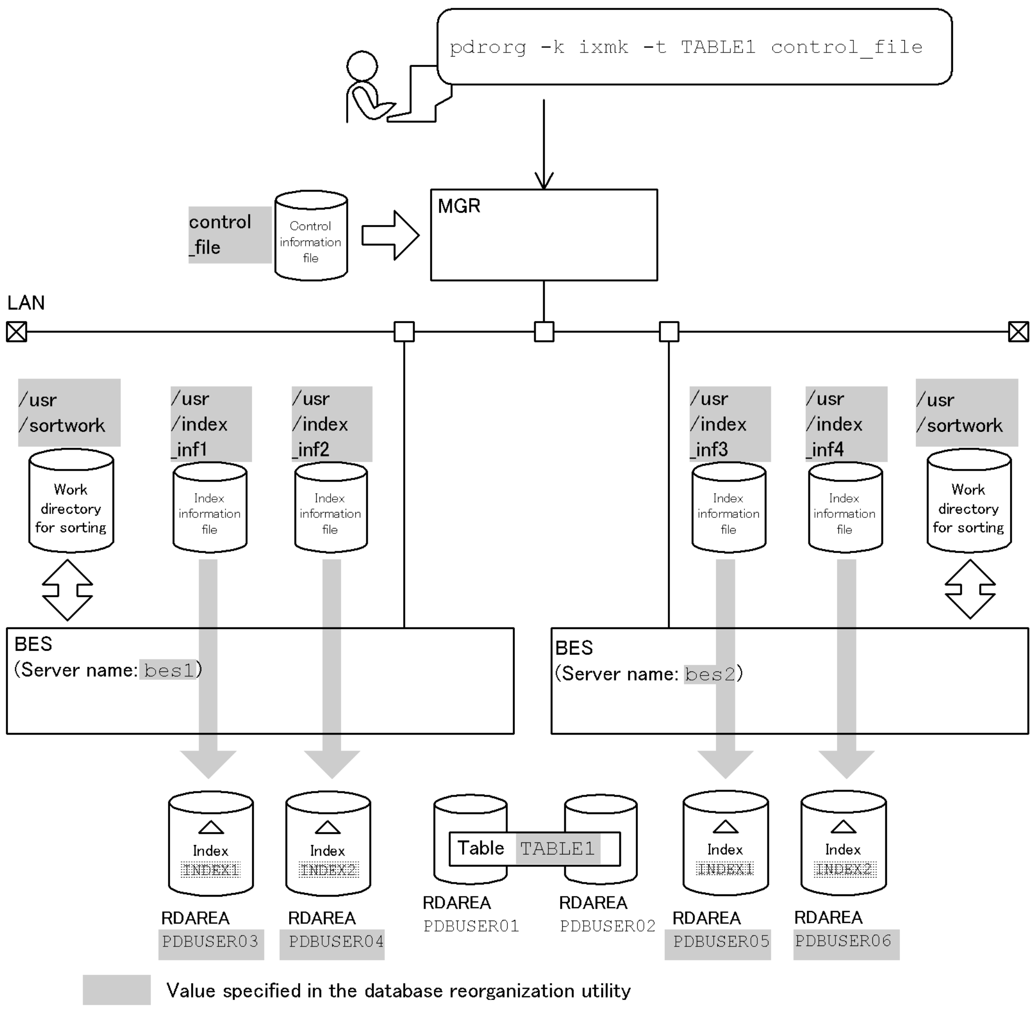
- Index definition (non-partitioning key index):

```
CREATE INDEX INDEX2 ON TABLE1(C2,C1)
               IN ((PDBUSER04),(PDBUSER06))
```

Overview



Relationship between input/output files and RDAREAs



Explanation of the command

This example creates indexes in batch mode on the basis of the index information output during data load operation.

`-k ixmk`: Specification for batch index creation

`-t TABLE1`: Name of the table subject to batch index creation

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
index INDEX1 PDBUSER03 /usr/index_inf1         1
index INDEX2 PDBUSER04 /usr/index_inf2         1
index INDEX1 PDBUSER05 /usr/index_inf3         1
index INDEX2 PDBUSER06 /usr/index_inf4         1
sort bes1 /usr/sortwork,8192                   2
sort bes2 /usr/sortwork,8192                   2
```

Explanation:

1. Specifies the index information files containing the index information:

   `INDEX1`, `INDEX2`: Index identifiers

   `PDBUSER03`, `PDBUSER04`, `PDBUSER05`, `PDBUSER06`: Names of the index storage RDAREAs

   `/usr/index_inf1`, `/usr/index_inf2`, `/usr/index_inf3`, `/usr/index_inf4`: Names of the index information files

2. Specifies the work directory for sorting:

   `bes1`, `bes2`: Names of the servers used to create the sort work file

   `/usr/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)

8.6.2 Cross-reference by purpose

Required options and control statements depend on the type of batch index creation.

The options listed in (1) and the control statements are mandatory. The items in (2) are related to the options and control statements presented in the *Reference* column. For details about the options, see Section 8.9.2.

(1) Required items

Item	Reference			
	Option		Control statement	
Type of `pdrorg` processing	`-k ixmk`	(1)	—	—
Name of the table to be subject to batch index creation	`-t`	(3)	—	—
File containing the control statements	*control-information-filename*	(19)	—	—

Item	Reference			
	Option		Control statement	
Information about the index information file	—	—	`index` statement	8.9.5

—: Not applicable

(2) Information to be specified depending on the batch index creation method

Item	Reference location			
	Option		Control statement	
Acquiring or not acquiring database update log	`-l`	(5)	—	—
Changing the `pdrorg` executor's user authorization identifier to a value other than the value of environment variable `PDUSER`[1]	`-u`	(10)	—	—
Executing batch index creation in batch input/output mode using local buffer instead of global buffer[2]	`-n`	(11)	—	—
Automatically deleting index information files after index creation	`-o`	(15)	—	—
Changing a batch index creation status message output interval to a value other than 100,000 lines	`-m`	(16)	—	—
Monitoring the response time for server-to-server communication	`-X`	(17)	—	—
Not enough space in the `/tmp` directory	—	—	`sort` statement	8.9.8
Monitoring the execution time of `pdrorg`	—	—	`option` statement	8.9.16
Changing the percentage of free space in index during batch index creation[3]	—	—	`option` statement	8.9.16
The inner replica facility is used	`-q`	(18)	—	—

—: Not applicable

[1] If omitted, the system assumes the value of the `PDUSER` environment variable. If the `PDUSER` environment variable has not been specified, the system assumes the user name in the login window.

[2] By specifying the number of batch input/output pages, you can reduce the number of I/O operations because data is input/output in units of specified pages in batch mode.

[3] If a space shortage occurs in an RDAREA during execution of `pdrorg`, you can complete the processing without having to expand the RDAREA temporarily (except for an index whose percentage of free space is 0).

8.7 Re-creating indexes

8.7.1 Examples

This section presents examples of using the database reorganization utility (re-creating indexes), listed as follows:

Example	Description	Classification
1	Re-creating a plug-in index • Re-creating a plug-in index defined for a table with an abstract data type (`SGMLTEXT`)	S
2	Re-creating indexes in units of index storage RDAREAs	P
3	Re-creating indexes in units of indexes	

S: HiRDB/Single Server

P: HiRDB/Parallel Server

(1) Re-creating a plug-in index

Example 1

This example re-creates a plug-in index (`INDEX1`) defined for a table (`TABLE1`) with columns of abstract data type (`SGMLTEXT`). The abstract data type (`SGMLTEXT`) is provided by the HiRDB Text Search Plug-in.

The example assumes that the following table and index have been defined:

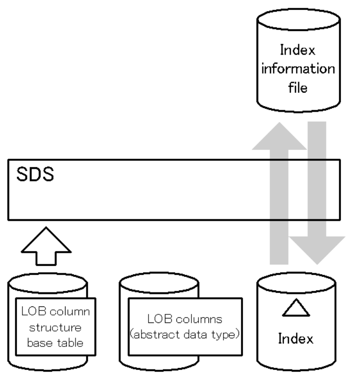
- Table definition:

```
CREATE TABLE TABLE1(C1 INT,C2 SGMLTEXT
         ALLOCATE (SGMLTEXT IN LOBUSER01)
         PLUGIN'<DTD>sgml.dtd</DTD>' ) IN PDBUSER01
```

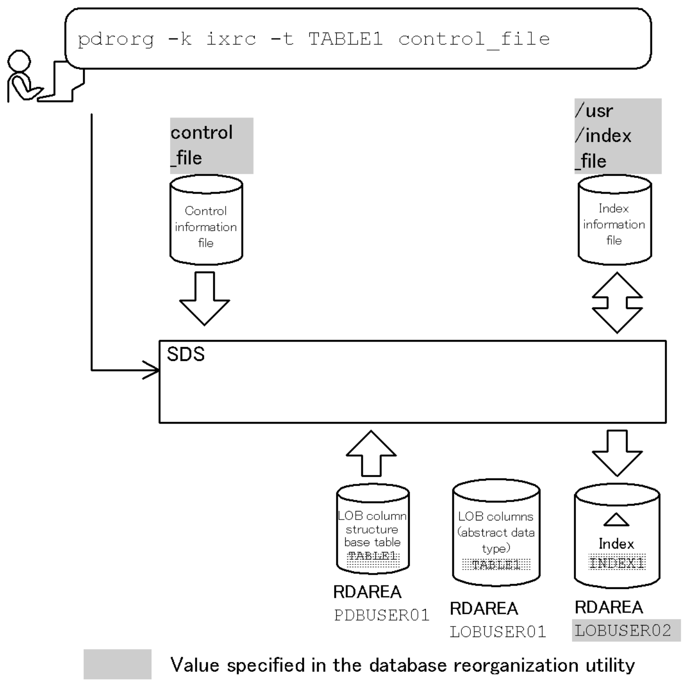
- Plug-in index definition:

```
CREATE INDEX INDEX1 USING TYPE NGRAM ON TABLE1(C2)
         IN LOBUSER02
```

Overview


Index information file
SDS
LOB column structure base table
LOB columns (abstract data type)
Index

Relationship between input/output files and RDAREAs



```
pdrorg -k ixrc -t TABLE1 control_file
```

Explanation of the command

This example creates a plug-in index (`INDEX1`) defined for a table (`TABLE1`).

`-k ixrc`: Specification for index re-creation

`-t TABLE1`: Name of the table subject to index re-creation

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
index INDEX1 LOBUSER02 /usr/index_file                    1
```

Explanation:

1. Specifies the index information file to which plug-in index information is to be output:

   `INDEX1`: Identifier of the plug-in index to be created

   `LOBUSER02`: Name of the RDAREA storing the plug-in index to be created

   `/usr/index_file`: Name of the index information file to which index information is output

(2) Re-creating an index in units of index storage RDAREAs

Example 2

This example re-creates an index (`INDEX1`) in units of RDAREAs (`PDBUSER03`).

The example assumes that the following table and indexes have been defined:

- Table definition:

```
CREATE TABLE TABLE1(C1 INT NOT NULL,C2 CHAR(8),C3 INT)
               IN ((PDBUSER01) C1 > 10,(PDBUSER02))
```

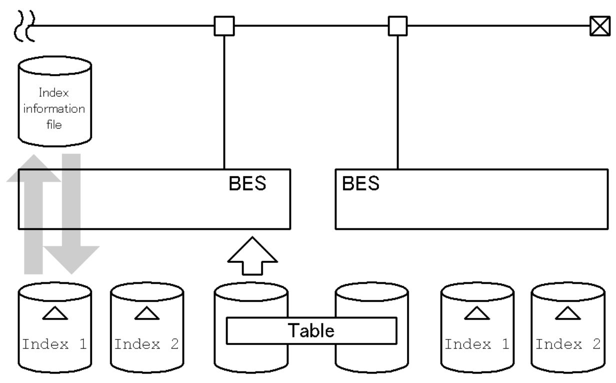
- Index definition (partitioning key index):

```
CREATE INDEX INDEX1 ON TABLE1(C1)
               IN ((PDBUSER03),(PDBUSER05))
```

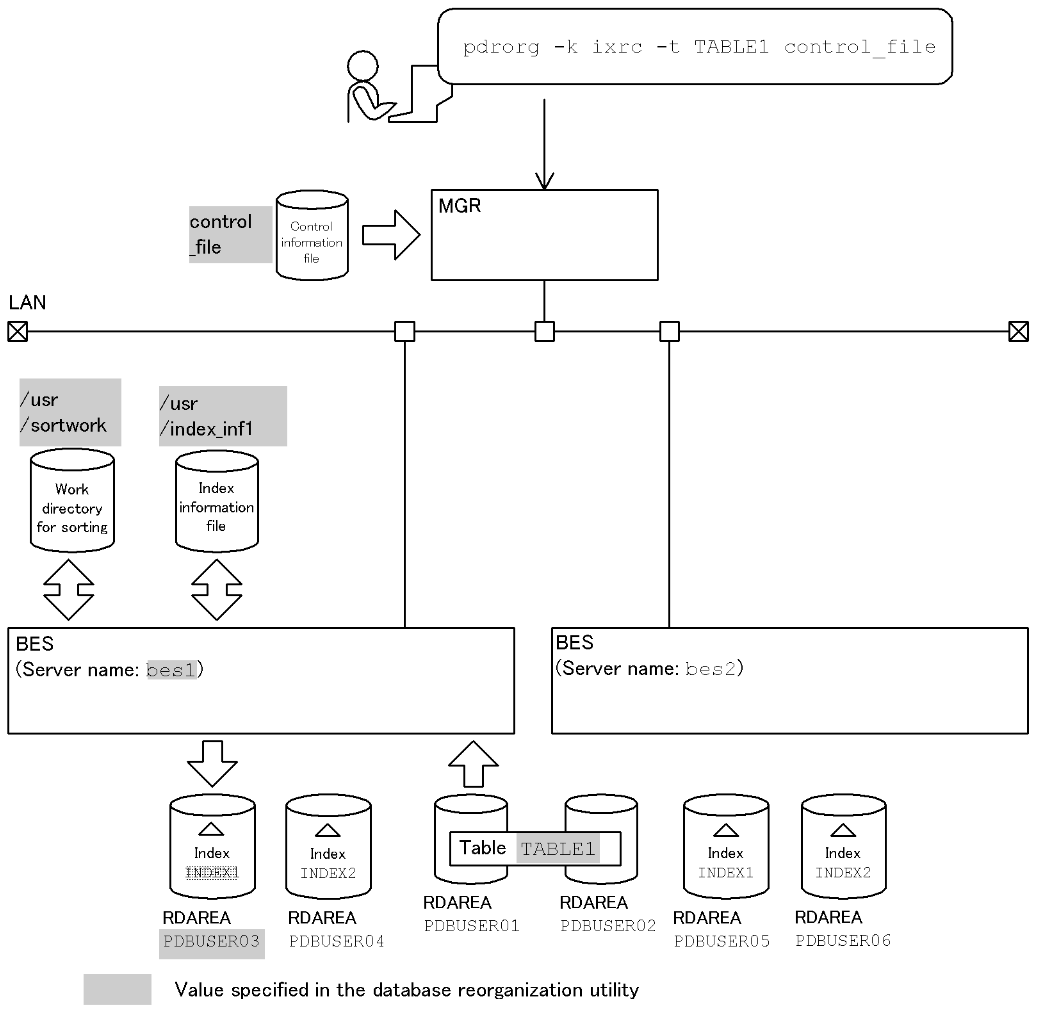
- Index definition (non-partitioning key index):

```
CREATE INDEX INDEX2 ON TABLE1(C2,C1)
               IN ((PDBUSER04),(PDBUSER06))
```

Overview


Index information file
BES
BES
Index 1
Index 2
Table
Index 1
Index 2

Relationship between input/output files and RDAREAs



Explanation of the command

This example re-creates indexes (`INDEX1` and `INDEX2`).

`-k ixrc`: Specification for index re-creation

`-t TABLE1`: Name of the table subject to index re-creation

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
index INDEX1 PDBUSER03 /usr/index_inf1              1
sort bes1 /usr/sortwork,8192                        2
```

Explanation:

1. Specifies the index information file:

   `INDEX1`: Index identifier

   `PDBUSER03`: Name of the index storage RDAREA

   `/usr/index_inf1`: Name of the index information file

2. Specifies the work directory for sorting:

   `bes1`: Name of the server used to create the sort work file

   `/usr/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)

(3) Re-creating indexes in units of indexes

Example 3

This example re-creates indexes (`INDEX1` and `INDEX2`) defined for a row-partitioned table (`TABLE1`).

The example assumes that the following table and indexes have been defined:

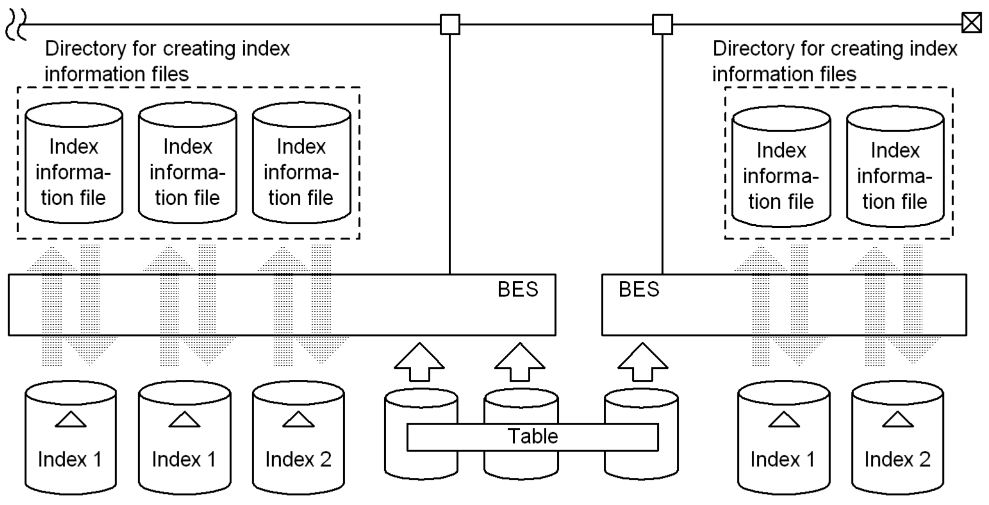
- Table definition:

```
CREATE TABLE TABLE1(C1 INT,C2 INT NOT NULL)
     IN ((PDBUSER01) C2<200,(PDBUSER02) C2<400,(PDBUSER03))
```

- Index definition:

```
CREATE INDEX INDEX1 ON TABLE1(C2) IN
((IDX01),(IDX02),(IDX03))
CREATE INDEX INDEX2 ON TABLE1(C1) IN ((IDX04),(IDX05))
```

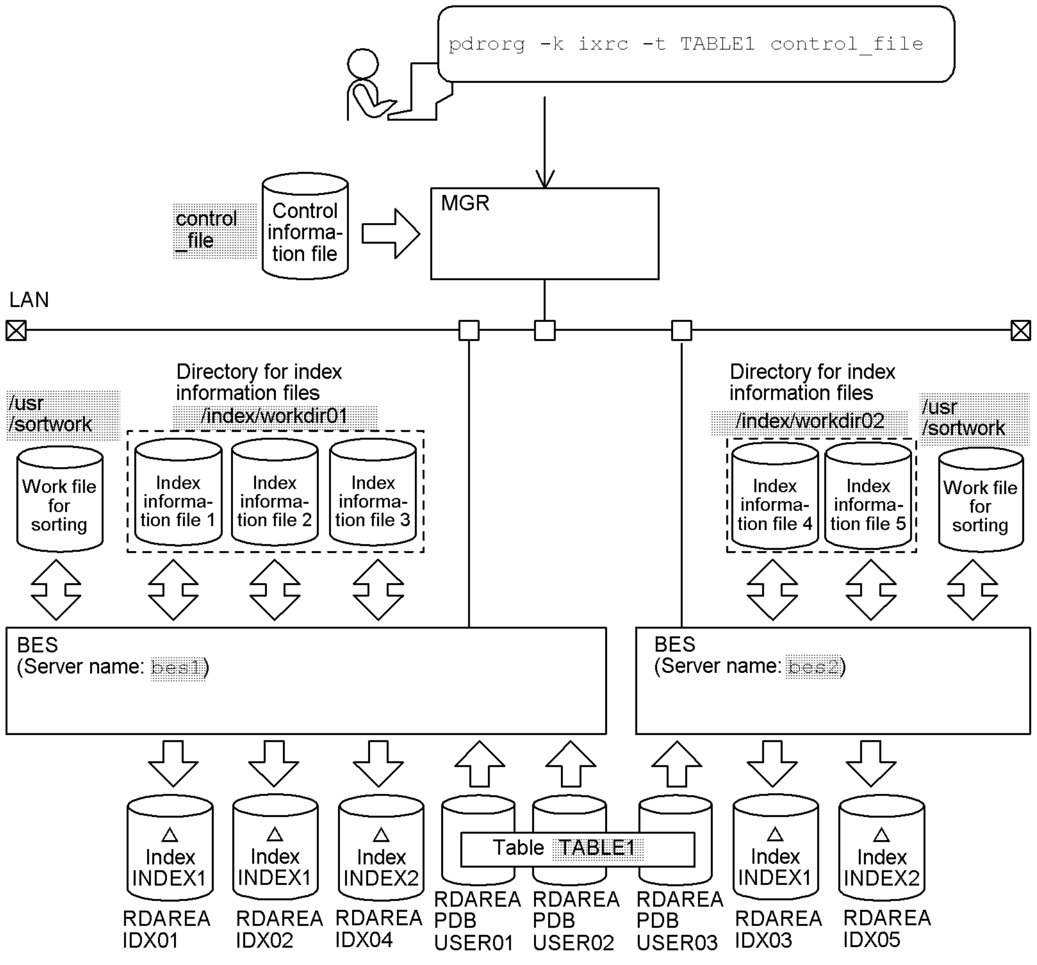
Overview



BES: Back-end server

(a) Re-creating all indexes defined for the table

Relationship between input/output files and RDAREAs



▨ : Value specified in the database reorganization utility
MGR: System manager
BES: Back-end server

Explanation of the command

This example re-creates indexes (`INDEX1` and `INDEX2`).

`-k ixrc`: Specification for index re-creation

`-t TABLE1`: Name of the table subject to index re-creation

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
idxname name=*                          1
idxwork bes1 /index/workdir01           2
idxwork bes2 /index/workdir02           2
sort bes1 /usr/sortwork,8192            3
sort bes2 /usr/sortwork,8192            3
```

Explanation:

1. Specifies index re-creation in units of indexes:

   `*`: Specification for re-creating all indexes defined for the table (`TABLE1`)

2. Specifies the directory in which index information files are to be created:

   `bes1`, `bes2`: Names of the servers used to create index information

   `/index/workdir01`, `/index/workdir02`: Names of the directories in which index information files are created

3. Specifies the work directory for sorting:

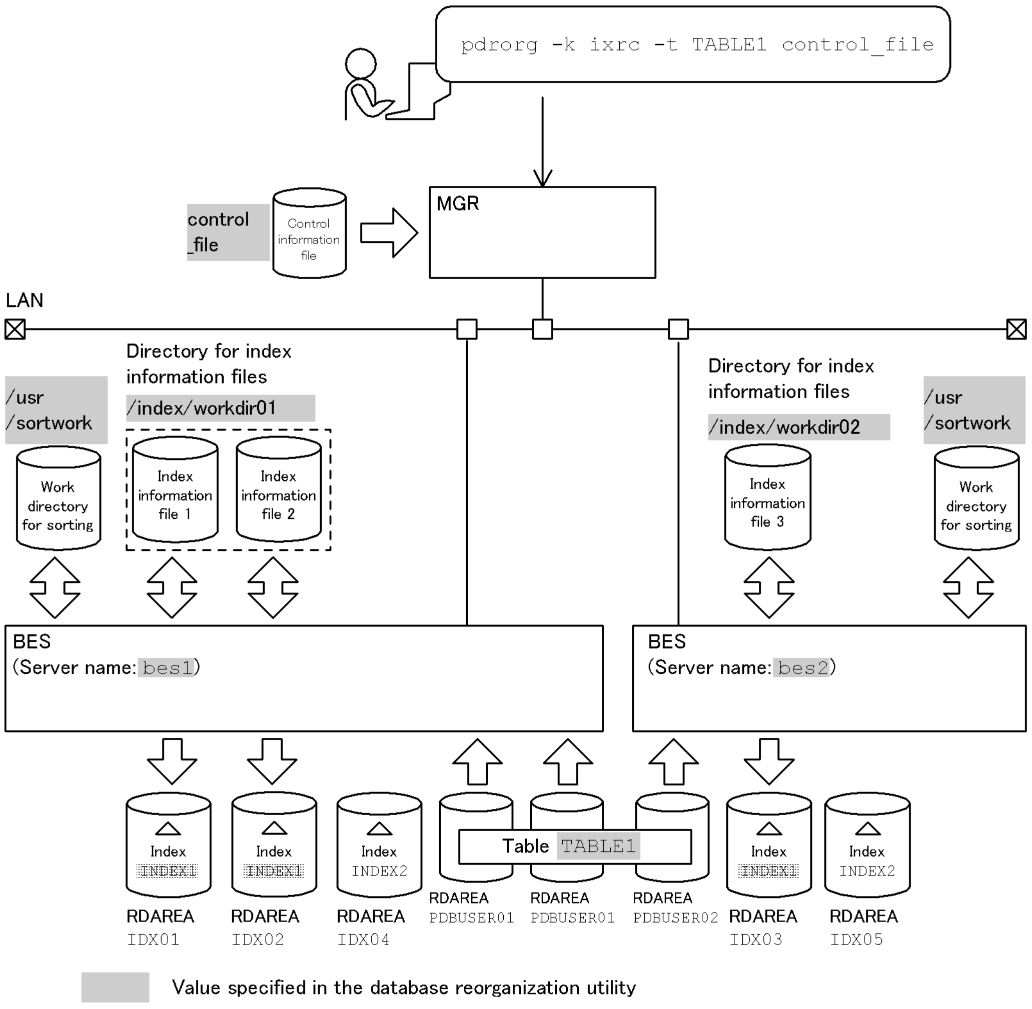
   `bes1`, `bes2`: Names of the servers used to create the sort work file

   `/usr/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)

(b) Re-creating one of the indexes defined for a table

Relationship between input/output files and RDAREAs



Explanation of the command

This example re-creates an index (`INDEX1`).

`-k ixrc`: Specification for index re-creation

`-t TABLE1`: Name of the table subject to index re-creation

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
idxname name=INDEX1                        1
idxwork bes1 /index/workdir01              2
idxwork bes2 /index/workdir02              2
sort bes1 /usr/sortwork,8192               3
sort bes2 /usr/sortwork,8192               3
```

Explanation:

1. Specifies index re-creation in units of indexes:

   `INDEX1`: Name of the index to be re-created

2. Specifies the directory in which index information files are to be created:

   `bes1`, `bes2`: Names of the servers used to create index information

   `/index/workdir01`, `/index/workdir02`: Names of the directories in which index information files are created

3. Specifies the work directory for sorting:

   `bes1`, `bes2`: Names of the servers used to create the sort work file

   `/usr/sortwork`: Name of the directory in which the sort work file is created

   `8192`: Size of buffer for sorting (in KB)

8.7.2 Cross-reference by purpose

Required options and control statements depend on the type of index re-creation.

The options listed in (1) and the control statements are mandatory. The items in (2) are related to the options and control statements presented in the *Reference* column. For details about the options, see Section 8.9.2.

(1) Required items

Item	Reference			
	Option		Control statement	
Type of `pdrorg` processing	`-k ixrc`	(1)	—	—
Name of the table subject to index re-creation	`-t`	(3)	—	—
File containing the control statements	*control-information-filename*	(19)	—	—

Item	Reference			
	Option		Control statement	
Information about the index information file or index	—	—	`index` or `idxname` statement	8.9.5 or 8.9.6

— : Not applicable

(2) Information to be specified depending on the index re-creation method

Item	Reference location			
	Option		Control statement	
Acquiring or not acquiring database update log	`-l`	(5)	—	—
Changing the `pdrorg` executor's user authorization identifier to a value other than the value of environment variable `PDUSER`1	`-u`	(10)	—	—
Re-creating in batch input/output mode using local buffer instead of global buffer2	`-n`	(11)	—	—
Automatically deleting index information files after index re-creation	`-o`	(15)	—	—
Changing an index re-creation status message output interval to a value other than 100,000 lines	`-m`	(16)	—	—
Monitoring the response time for server-to-server communication	`-X`	(17)	—	—
Not enough space in the `/tmp` directory when index is re-created in units of index storage RDAREAs (using the `index` statement)	—	—	`sort` statement	8.9.8
Not enough space in the `/tmp` directory when index is re-created in units of indexes (using the `idxname` statement)	—	—	`idxwork` statement	8.9.7
			`sort` statement	8.9.8
Monitoring the execution time of `pdrorg`	—	—	`option` statement	8.9.16
Changing the percentage of free space in index during index re-creation3	—	—	`option` statement	8.9.16
The inner replica facility is used	`-q`	(18)	—	—

— : Not applicable

1 If omitted, the system assumes the value of the `PDUSER` environment variable. If the `PDUSER` environment variable has not been specified, the system assumes the user

name in the login window.

2 By specifying the number of batch input/output pages, you can reduce the number of I/O operations because data is input/output in units of specified pages in batch mode.

3 If a space shortage occurs in an RDAREA during execution of `pdrorg`, you can complete the processing without having to expand the RDAREA temporarily (except for an index whose percentage of free space is 0).

8.8 Reorganizing indexes

8.8.1 Example

This section presents an example of using the database reorganization utility (reorganizing indexes).

Example 1

This example reorganizes the indexes (`INDEX1` and `INDEX2`) defined for a table (`TABLE1`).

The example assumes that the following table and indexes have been defined:

- Table definition:

```
CREATE TABLE TABLE1(C1 INT NOT NULL,C2 CHAR(8),C3 INT)
               IN ((PDBUSER01) C1 > 10,(PDBUSER02))
```

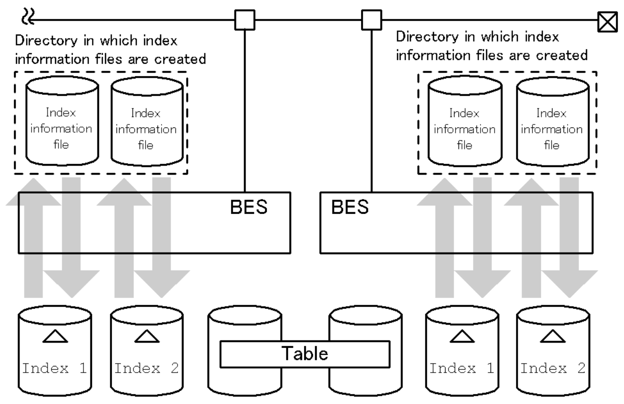
- Index definition (partitioning key index):

```
CREATE INDEX INDEX1 ON TABLE1(C1)
               IN ((PDBUSER03),(PDBUSER05))
```

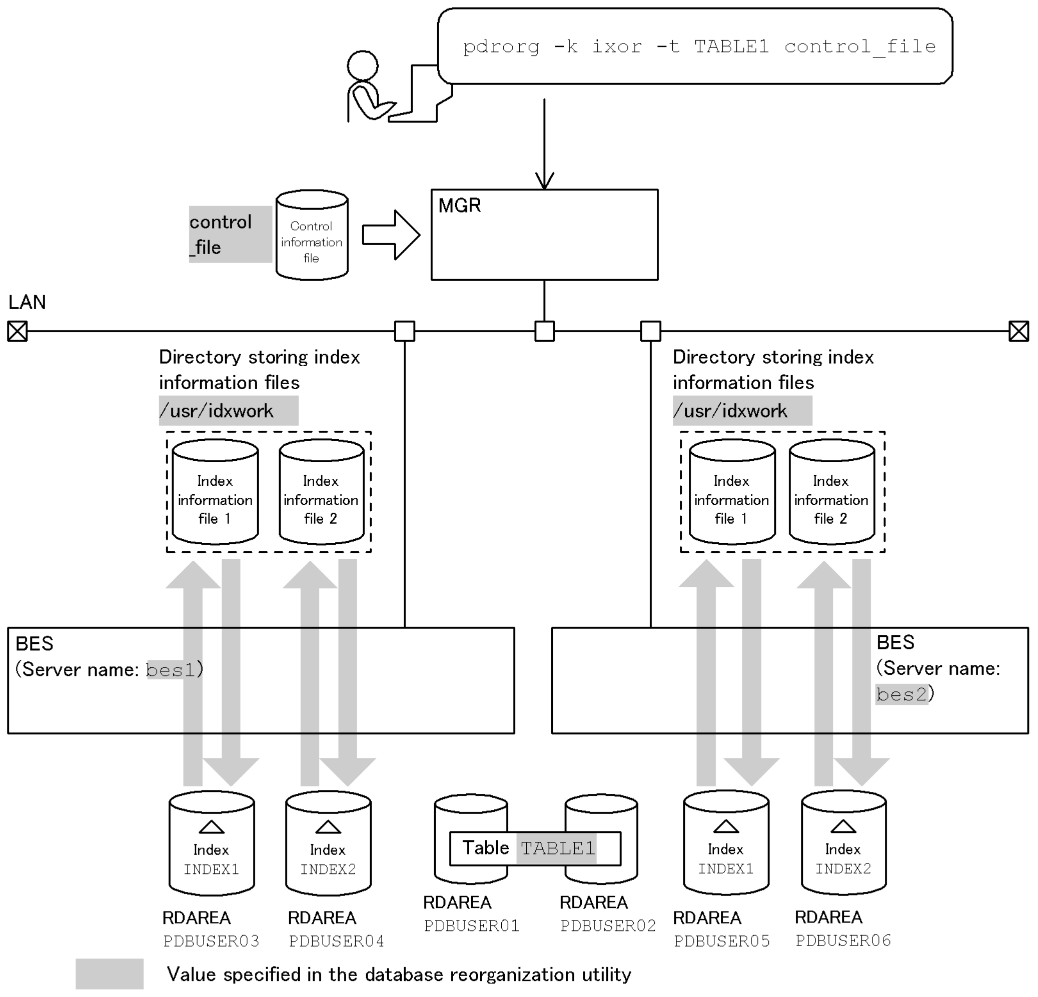
- Index definition (non-partitioning key index):

```
CREATE INDEX INDEX2 ON TABLE1(C2,C1)
               IN ((PDBUSER04),(PDBUSER06))
```

Overview



Relationship between input/output files and RDAREAs



Explanation of the command

This example reorganizes the indexes (`INDEX1` and `INDEX2`) defined for a table (`TABLE1`).

`-k ixor`: Specification for index reorganization

`-t TABLE1`: Name of the table subject to index reorganization

`control_file`: Name of the control information file

Contents of the control information file (control_file)

```
idxname name=*                        1
idxwork bes1 /usr/idx_file            2
idxwork bes2 /usr/idx_file            2
```

Explanation:

1. Specifies index reorganization in units of indexes:

   `*`: Specification for reorganizing all indexes defined for the table (`TABLE1`)

2. Specifies the directory for index information files:

   `bes1`, `bes2`: Names of the servers used to create index information

   `/usr/idx_file`: Name of the directory in which index information files are created

8.8.2 Cross-reference by purpose

Required options and control statements depend on the type of index reorganization.

The options listed in (1) and the control statements are mandatory. The items in (2) are related to the options and control statements presented in the *Reference* column. For details about the options, see Section 8.9.2.

(1) Required items

Item	Reference			
	Option		Control statement	
Type of `pdrorg` processing	`-k ixor`	(1)	—	—
Name of the table subject to index reorganization	`-t`	(3)	—	—
File containing the control statements	*control-information-filename*	(19)	—	—
Information about the index information file or index	—	—	`index` or `idxname` statement	8.9.5 or 8.9.6

—: Not applicable

(2) Information to be specified depending on the index reorganization method

Item	Reference location			
	Option		Control statement	
Acquiring or not acquiring database update log	`-l`	(5)	—	—
Changing the `pdrorg` executor's user authorization identifier to a value other than the value of environment variable `PDUSER`[1]	`-u`	(10)	—	—
Automatically deleting index information files after index reorganization	`-o`	(15)	—	—
Monitoring the response time for server-to-server communication	`-X`	(17)	—	—
Not enough space in the `/tmp` directory when index is reorganized in units of indexes (using the `idxname` statement)	—	—	`idxwork` statement	8.9.7
Monitoring the execution time of `pdrorg`	—	—	`option` statement	8.9.16
Changing the percentage of free space in index during index reorganization[2]	—	—	`option` statement	8.9.16
The inner replica facility is used	`-q`	(18)	—	—

—: Not applicable

[1] If omitted, the system assumes the value of the `PDUSER` environment variable. If the `PDUSER` environment variable has not been specified, the system assumes the user name in the login window.

[2] If a space shortage occurs in an RDAREA during execution of `pdrorg`, you can complete the processing without having to expand the RDAREA temporarily (except for an index whose percentage of free space is 0).

8.9 Command format

8.9.1 Format

This section explains the format of the `pdrorg` command. In the following table, each number corresponds to the number assigned to each option.

Options in bold are important or mandatory.

No.	Format
1	`pdrorg` **`-k`** ***processing-type***
2	`[-c` *object-of-processing*`]`

3	**`[-t {[`*authorization-identifier*`.]`*table-identifier*`|[`*authorization-identifier*`.]all`**
	`|`*table-identifier*`[,`*table-identifier*`]...}]`

4	`[-r` *RDAREA-name*`]`
5	**`[-l`** ***log-acquisition-method*`]`**
6	**`[-i`** ***index-creation-method*`]`**

7	`[-W {{dat|extdat}[,[`*separator-character*`][,{cr|crlf}][,sup]]`
	`|bin`
	`|fixtext[,[`*padding-character*`][,{cr|crlf}]]}]`

8	`[-g]`
9	`[-j]`
10	`[-u` *authorization-identifier*`]`

11	`[-n [`*batch-input/output-local-buffer-sectors-count*`],[div],`
	`[`*random-access-local-buffer-sectors-count*`]]`

12	`[-f` *unload-data-file-type-or-LOB-data-unload-file-type*`]`
13	`[-b` *unload-sequence*`]`
14	`[-s]`
15	`[-o]`
16	`[-m` *progress-message-output-interval*`]`
17	`[-X` *response-monitoring-time-for-server-to-server-communication*`]`
18	`[-q` *generation-number*`]`

No.	Format
19	***control-information-filename***

Note

Be sure to specify *control-information-filename* as the last option.

Relationships between functions and options of pdrorg

The following table shows the relationships between the functions and options of the `pdrorg` command:

Options	pdorg's functions						
	Table reorg	Table unldg	Table reldg	Data dict tbl reorg	Batch index creation	Index re-creati on	Index reorg
`-k`	R	R	R	R	R	R	R
`-c`	O	O	O	R	—	—	—
`-t`	R	R	R	O	R	R	R
`-r`	O	O	O	—	—	—	—
`-l`	O	—	O	O	O	O	O
`-i`	O	—	O	—	—	—	—
`-W`	—	O	—	—	—	—	—
`-g`	O	O	O	—	—	—	—
`-j`	O	O	O	—	—	—	—
`-u`	O	O	O	O	O	O	O
`-n`	O	O	O	—	O	O	—
`-f`	O	O	O	O	—	—	—
`-b`	O	O	—	—	—	—	—
`-S`	O	O	—	—	—	—	—
`-o`	O	—	O	—	O	O	O
`-m`	O	O	O	O	O	O	—
`-X`	O	O	O	O	O	O	O
`-q`	O	O	O	—	O	O	O

Options	pdorg's functions						
	Table reorg	**Table unldg**	**Table reldg**	**Data dict tbl reorg**	**Batch index creation**	**Index re-creati on**	**Index reorg**
control-informa tion-filename	R	R	R	R	R	R	R

R: Required

O: Optional

—: Not specifiable

8.9.2 Options

(1) -k processing-type

Specifies the type of processing to be executed by `pdrorg`:

`rorg`

Reorganize a table.

`unld`

Unload a table.

`reld`

Reload a table.

`ixmk`

Execute batch index creation.

`ixrc`

Re-create an index.

`ixor`

Reorganize an index.

(a) Notes

When `-k rorg` is specified, the utility executes unload, reload, and index creation processing (if `-i c` specified) sequentially in this order. If the utility terminates abnormally during the processing, the utility stores information about the processing that resulted in abnormal termination as a status in the database table. Therefore, when the same processing is restarted, the utility starts from the processing that resulted in abnormal termination. This means that the utility does not always start with unload processing. For details, see *8.4.4 Handling of a reloading error during table* *reorganization*.

To always start with unload processing, regardless of whether the previous `pdrorg` terminated normally or abnormally, consider using the type of operation that executes `-k unld` and `-k reld` separately.

(2) -c object-of-processing

～ <<`user`>>

Specifies the type of table to be reorganized:

`dic`

Reorganize a data dictionary table.

`user`

Reorganize a table.

(3) -t {[authorization-identifier.]table-identifier|[authorization-identifier.]all|table-identifier[,table-identifier]...}

Specifies the name of the table being processed.

(a) User-defined table

To specify a user-defined table, use the following format:

`-t {[`*authorization-identifier*`.]`*table-identifier*| `[`*authorization-identifier*`.]all}`

Specify either only one table or `all` to reorganize all tables owned by the schema. Reorganizing all tables owned by a schema is called reorganizing in units of schemas.

If you omit the authorization identifier, the system assumes the user name used to establish connection with HiRDB.

Rules

1. You cannot specify a view table.
2. If an authentication identifier or table identifier is enclosed in double quotation marks (`"`), the command treats it as being case sensitive. If it is not enclosed in double quotation marks (`"`), the command treats it as in all uppercase letters. If you are using `sh` (Bourne shell), `csh` (C shell), or `ksh` (Korn shell), you must enclose the entire identifier in single quotation marks (`'`).

Rules for reorganization in units of schemas

1. Reorganization in units of schemas is applicable to `-k rorg`, `-k unld`, and `-k reld`, that does not involve reorganization or reloading with the synchronization point specification.

2. If you unload a table in units of schemas, you can use the obtained unload data file to reload the table in units of tables. On the other hand, if you unload a table in units of tables, you cannot use the obtained unload data file to reload the table in units of schemas.
3. Tables are unloaded or reloaded in the order that their table identifiers are specified.
4. If an error occurs while processing a table, the system terminates the processing at that point. If an error occurs while reloading a table, reload the affected table as well as any unprocessed table in units of tables. Reorganization has been completed for the tables for which processing has terminated normally.
5. If you are reorganizing a table in units of schemas, use the `pdhold` command to shut down the RDAREAs storing the table in the schema so that the UAP that accesses the table under reorganization is not placed in wait status.
6. If tables contained in the schema are different between unload and reload operations, the system handles them as follows:

   Tables during unload operation: (`T1`, `T3`, `T5`, `T7`)

   Tables during reload operation: (`T1`, `T2`, `T3`, `T5`)

   `T1`, `T3`, `T5`: Are reloaded.

   `T2`: Is skipped because it was not in the schema during unload operation.

   `T7`: Is not reloaded because it was not in the schema during unload operation.
7. If there is an external table for the schema, the external table will not be processed.

(b) Data dictionary tables

To specify data dictionary tables, use the following format:

`-t` *table-identifier*`[,`*table-identifier*`]...`

You can specify as many table identifiers as there are data dictionary tables. You cannot specify an authorization identifier. To reorganize all data dictionary tables, omit the `-t` option.

Before starting reorganization of a data dictionary table, make a backup copy of the data dictionary RDAREA (if a space shortage occurs in the RDAREA during reload processing, the backup copy can be used to restore the RDAREA to its status before reorganization and then you can expand the RDAREA).

You can specify the following data dictionary table identifiers:

No.	Table identifier	Remarks
1	`SQL_PHYSICAL_FILES`	—
2	`SQL_RDAREAS`	
3	`SQL_TABLES`	
4	`SQL_COLUMNS`	
5	`SQL_INDEXES`	
6	`SQL_USERS`	
7	`SQL_RDAREA_PRIVILEGES`	
8	`SQL_TABLE_PRIVILEGES`	
9	`SQL_DIV_TABLE`	
10	`SQL_DIV_TABLE_REGULARIZE`	Table used by the system
11	`SQL_INDEX_COLINF`	—
12	`SQL_TABLE_STATISTICS`	
13	`SQL_COLUMN_STATISTICS`	
14	`SQL_INDEX_STATISTICS`	
15	`SQL_VIEW_TABLE_USAGE`	
16	`SQL_VIEWS`	
17	`SQL_VIEW_DEF`	Table used by the system
18	`SQL_DIV_INDEX`	—
19	`SQL_DIV_COLUMN`	
20	`SQL_REFERENTIAL_CONSTRAINTS`	Table used by the system

No.	Table identifier	Remarks
21	`SQL_ALIASES`	--
22	`SQL_ROUTINES`	
23	`SQL_ROUTINE_RESOURCES`	
24	`SQL_ROUTINE_PARAMS`	
25	`SQL_DATATYPES`	
26	`SQL_DATATYPE_DESCRIPTORS`	
27	`SQL_TABLE_RESOURCES`	
28	`SQL_PLUGINS`	
29	`SQL_PLUGIN_ROUTINES`	
30	`SQL_PLUGIN_ROUTINE_PARAMS`	
31	`SQL_REGISTRY_CONTEXT`	Table used by the system
32	`SQL_REGISTRY_KEY`	
33	`SQL_INDEX_TYPES`	None
34	`SQL_INDEX_RESOURCES`	
35	`SQL_INDEX_DATATYPE`	
36	`SQL_INDEX_FUNCTION`	
37	`SQL_TYPE_RESOURCES`	
38	`SQL_INDEX_TYPE_FUNCTION`	
39	`SQL_EXCEPT`	
40	`SQL_FOREIGN_SERVERS`	
41	`SQL_USER_MAPPINGS`	
42	`SQL_USAGE_PRIVILEGES`	Table used by the system
43	`SQL_IOS_GENERATIONS`	None
44	`SQL_PARTKEY`	
45	`SQL_PARTKEY_DIVISION`	
46	`SQL_AUDITS`	
47	`SQL_AUDIT_REGULARIZE`	Table used by the system

No.	Table identifier	Remarks
48	`SQL_TRIGGERS`	None
49	`SQL_TRIGGER_ACTCOND`	Table used by the system
50	`SQL_TRIGGER_COLUMNS`	None
51	`SQL_TRIGGER_DEF_SOURCE`	
52	`SQL_TRIGGER_USAGE`	
53	`SQL_KEYCOLUMN_USAGE`	
54	`SQL_TABLE_CONSTRAINTS`	
55	`SQL_CHECKS`	
56	`SQL_CHECK_COLUMNS`	
57	`SQL_REFERENTIAL_CONSTRAINTS`	
58	`SQL_DIV_TYPE`	
59	`SQL_SYSPARAMS`	
60	`SQL_DB_MANAGEMENT`	Table used by the system
61	`SQL_DB_STATE_ANALYZED`	
62	`SQL_PUBLICVIEW_SAME_USERS`	

(4) -r RDAREA-name

~ <identifier> ((1-30))

For a row-partitioned table stored in multiple RDAREAs, this option specifies the name of the RDAREA being reorganized.

When this option is omitted, the utility assumes that all data for the row-partitioned table is being reorganized.

(a) Rules

1. If an RDAREA name is enclosed in double quotation marks (`"`), the command treats it as being case sensitive. If it is not enclosed in double quotation marks (`"`), the command treats it as in all uppercase letters. If an RDAREA name contains a space, enclose the RDAREA name in double quotation marks (`"`). If you are using `sh` (Bourne shell), `csh` (C shell), or `ksh` (Korn shell), you must enclose the entire set of RDAREA names in single quotation marks (`'`).
2. If you have unloaded a table partitioned by the flexible hash partitioning method in units of RDAREAs and changed the hash function or added an RDAREA, the

system ignores the hash function during the reload operation and stores the data as is in the original RDAREAs.

If you have unloaded a table partitioned by the key range or FIX hash partitioning method in units of RDAREAs, you cannot reload the table using different partitioning conditions.

To reorganize a table by changing the hash function, adding an RDAREA, or modifying the partitioning conditions, use the method of reorganization in units of tables, specifying the `-g` option. To change the hash function or add an RDAREA, use the `ALTER TABLE` definition SQL statement. To determine whether or not reloading is permitted when the table definition for an unload table does not match the table definition of the reload table, see Section *8.4.3 Whether or not reloading is permitted when the table definitions of unload table and reload table do not match*.

3. When reorganizing tables in units of schemas, you cannot specify the `-r` option.
4. If you are specifying a replica RDAREA, specify its original RDAREA name in this option and also specify the target generation number in the `-q` option.

(5) -l log-acquisition-method

~ <<p>>

Specifies the method for acquiring the database update log during the execution of `pdrorg`.

If `pdrorg` terminates abnormally during execution, the database is not restored to its status immediately before execution of `pdrorg` even when a database update log has been acquired. For details about how to handle abnormal termination of `pdrorg`, see *8.13 Database status in the event of an error and recovery method*.

`a`

Indicates the log acquisition mode in which the system collects database updating log information required for rollback and rollforward.

Criteria

This mode is suitable for reorganizing a small amount of table data. When `pdrorg` is executed in the log acquisition mode, there is no need to make backups either before or after execution of the utility, but performance is lower than in the other modes.

`p`

Indicates the pre-update log acquisition mode in which the system collects database updating log information required for rollback, but not the database updating log information required for rollforward.

Criteria

This mode is suitable for reorganizing a large amount of table data. When `pdrorg` is executed in pre-update log acquisition mode, execution time is faster than in the log acquisition mode. However, you need to make backup copies after the execution of `pdrorg` to protect against possible media errors.

If an error occurs during execution of `pdrorg`, `pdrorg` restores the database to the synchronization point immediately preceding the error. This prevents RDAREAs from being placed in no-log shutdown status; however, the target table is not restored to its status immediately before execution of `pdrorg`.

`n`

Indicates the no-log mode in which the system does not collect database updating log information.

Criteria

When the `pdrorg` command is executed in the no-log mode, execution time is faster than in other modes. However, you must make backups before and after execution of the `pdrorg` command so that the database can be restored from its backup in the event of an error during utility execution or from its backup and log information in the event of a media error. This mode is suitable for reorganizing a large amount of table data.

Notes

1. If an error occurs while `pdrorg` is executing in the no-log mode, you need to either restore the RDAREA that was shut down due to an error from its backup copy or re-initialize it.
2. You cannot specify the no-log mode for a reorganization or reload operation with the synchronization point specification.
3. For index re-creation (`-k ixrc`), the system assumes the log acquisition mode to protect the database under key search processing from being placed in error shutdown status. For index reorganization (`-k ixor`), the system assumes the pre-update log acquisition mode for the same reason. Therefore, the segment release log is output during key search processing, even if you specify the no-log mode.

(a) Notes

1. If `p` or `n` is specified and the database can be restored from its previous backup copy and log information or from its unload data (applicable if the corresponding RDAREA contains only the table being reorganized), there is no need to make backup copies before the execution of `pdrorg`.
2. For details about the operating method when `p` or `n` is specified (when the

database updating log information is not collected), see the *HiRDB Version 8 System Operation Guide*.

3. The transaction log is always collected by the transaction (`T`) that is generated by `pdrorg`, whether or not the `-l` option is specified. The system creates the following amount of transaction log information per server; the formula for determining the amount of log information during execution of `pdrorg` is shown below:

   Amount of log = $(1328 + 176 \times 3) \times T + A$ (bytes)

   `-k unld` specified:

   $T = (x \times 2) + (y \times 2)$

   `-k reld` specified:

   $T = (x \times 3) + (y \times 3) + (z \times 2)$

   `-k ixrc` specified:

   $T = (x \times 2) + (z \times 2)$

   `-k ixmk` specified:

   $T = z \times 2$

   `-k ixor` specified:

   $T = z \times 3$

   `-k rorg` specified:

   T = Value when `-k unld` is specified + value when `-k reld` is specified + $(x + y)$

   x: Number of tables (for reloading with synchronization point specification, the number of synchronization points)*

   y: Number of LOB column (LOB attribute) storage RDAREAs (when `-j` is specified, 1)*

   z: Number of indexes × number of index storage RDAREAs (this is not needed if `-i s` is specified)*

   * For processing in units of schemas, you must determine the value for all tables owned by the corresponding schema.

   A: Amount of system log information that is output according to database manipulation (amount of database update log information). This value depends on the value of the `-l` option. For details about determining the amount of system log information, see the *HiRDB Version 8 Installation and Design Guide*.

If `p` or `n` is specified, the system collects the following amount of `ENQ` log information per server by lock control:

`ENQ` log = $(p + q + r) \times T$

p: *Number* of table storage RDAREAs

q: Number of LOB column (LOB attribute) storage RDAREAs

r: Number of index storage RDAREAs

Therefore, the system outputs as much system log information as equals record length of system log file × `ENQ` log information.

4. When `-k unld` is specified, the log acquisition mode (`-l a`) is assumed regardless of the `-l` option value.
5. If you use Real Time SAN Replication based on the log-only synchronous method and have executed `pdrorg` with `-l p` or `-l n` specified at the transaction execution site, you must execute the preparations for log application. For details about the preparations for log application, see the manual *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide*.

(6) -i index-creation-method

Specifies the index creation method.

When this option is omitted, the utility assumes `c` for a user-defined table and `s` for a data dictionary table.

`c`

Indicates the batch index creation mode in which the system executes batch index creation immediately after reloading the table.

Criterion

When you are reorganizing a large amount of table data, this mode enables you to create an index at high speed.

`n`

Indicates the index information output mode in which the system outputs only index information to the index information file.

Criterion

You can create an index at high speed by executing multiple batch index creation (`-k ixmk`) processes concurrently by `pdrorg` using the index information files that have been output. This method is especially effective for a table that is partitioned in multiple servers constituting a HiRDB/ Parallel Server.

Notes

1. If you specify the index information output mode, you need to execute batch index creation (`-k ixmk`) with `pdrorg` after reloading data to the table. If you use `ALTER TABLE` to add an RDAREA without having executed batch index creation, the index information file obtained is not usable. To add an RDAREA using `ALTER TABLE`, be sure to complete batch index creation beforehand. If you have added an RDAREA with `ALTER TABLE`, you need to execute index re-creation (`-k ixrc`) with `pdrorg`.
2. If a plug-in index is defined for a table being reorganized, you cannot specify `-i n` unless the plug-in supports the batch plug-in index creation partial recovery facility.

`s`

Indicates the index update mode in which the system updates the index each time a row is stored.

Criterion

Specify this option when reorganizing a table with a small amount of data.

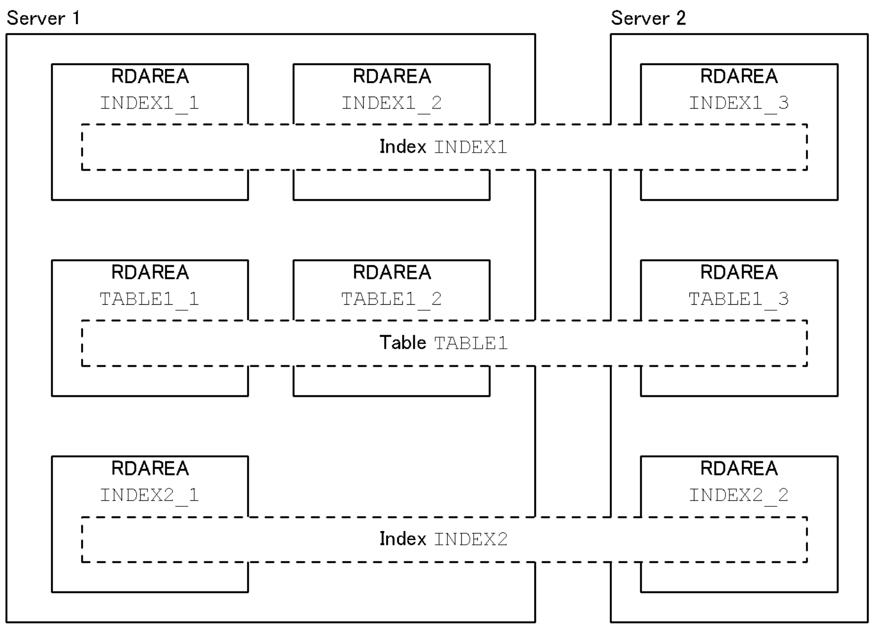
(a) Notes about index creation

1. Figure 8-16 shows the method for creating a row-partitioned index and non-partitioned index. If a table is partitioned and stored in multiple RDAREAs at the same server, a row-partitioned index is stored in as many index storage RDAREAs as there are table storage RDAREAs, while a non-partitioned index is always stored in a single index storage RDAREA, regardless of the number of table storage RDAREAs.

   A row-partitioned index is stored in the same manner as for a table that is partitioned within the same server (RDAREAs `INDEX1_1` and `INDEX1_2` in the figure).

   A non-partitioned index is always stored in a single RDAREA in the server regardless of the number of table storage RDAREAs (RDAREAs `INDEX1_3`, `INDEX2_1`, and `INDEX2_2` in the figure).

Figure 8-16: Index creation method for a partitioning key index and non-partitioning key index



Explanation:

Index creation depends on the `-i` option during table reorganization. For example, if you reorganize `TABLE1_1` in units of RDAREAs, row-partitioned index `INDEX1_1` is created (there is no effect on `INDEX1_2`). In this case, non-partitioned index `INDEX2_1` is not created because it requires information about both `TABLE1_1` and `TABLE1_2` (the index information file is created only for `TABLE1_1`).

To create a non-partitioned index, you also need to reorganize `TABLE1_2` to create the index information file for this table, then execute `pdrorg`'s batch index creation using the index information files for `TABLE1_1` and `TABLE1_2`.

2. After the reload operation is completed, the corresponding index is placed in unfinished status, thereby unusable until batch index creation is completed.

3. If you specify `-i c` or `-i n`, the command creates as many index information files as the *number of indexes* × *number of table storage RDAREAs*. Because these files are opened simultaneously, the maximum number of files permitted per process may be exceeded. If this is the case, increase the `pd_max_open_fds` operand value in the system definition. If the value of the `pd_max_open_fds` operand is exceeded, check and, if necessary, revise the number of table partitions

per server and the number of defined indexes, or specify `-i s`.

(b) Notes about specifying -i c or -i n

If `-i c` or `-i n` is specified and the `index` and `idxwork` statements are omitted, the system outputs the index information file to the `/tmp` directory using the following naming convention:

`/tmp/INDEX-`*index-name*`-`*index-storage-RDAREA-name*`-`*unique-character-string*

If `pdrorg` terminates abnormally, this file is not deleted. If you re-execute `pdrorg`, another index information file is created under a different name. Because this may result in a space shortage in the `/tmp` directory, you should delete unneeded index information files using the OS's `rm` command.

(7) -W {{dat|extdat}[,[separator-character][,{cr|crlf}][,sup]]|bin|fixtext[,[padding-character][,{cr|crlf}]]}

Specifies that the unload data file is used as the input data file to the database load utility.

When this option is specified, the system outputs table data to an unload data file in DAT, extended DAT, binary, or fixed-size data format.

For details about the format of the unload data file that is output by `pdrorg` specifying the `-W` option, see Section *8.3.3 Format of database load utility input files*.

`dat`

Specifies that data is to be output in the DAT format.

`extdat`

Specifies that data is to be output in the extended DAT format.

When you output data in extended DAT format, you cannot specify more than one unload data file.

separator-character ~ <character string>

Specifies that the separator between data items is to be changed to a character other than the comma (`,`) when data is output in DAT or extended DAT format.

When this option is omitted, the utility assumes a comma as the separator character.

Rules

1. You cannot specify any of the following characters because they are not permitted as separator characters for the input data file to the database

load utility:

`*` (asterisk), `"` (double quotation mark), _ (underline)

2. The following characters, which can occur in data being unloaded, are not suitable for use as separator characters:

   - Uppercase letters (A-Z) and lowercase letters (a-z)
   - Numeric characters (0-9)
   - Characters that can conflict with the results of conversion of row data into a DAT format
   - Characters that can occur as character codes used in Japanese-language input:

     | \ [ ] ( ) { } ‾ (overbar)
   - Signs for numeric data (+ or -)
   - Hyphen (-) for date data input
   - Colon (:) for time data input
   - Period (.) for time and date interval data input

3. To use the comma as the separator character, do not specify a separator character.

4. If the separator character contains a blank, enclose the entire separator character in double quotation marks.

`{cr|crlf}`

Specifies the linefeed code to be used for data output in DAT or extended DAT format. When this option is omitted, the utility assumes `cr`.

`cr`: Outputs `0x0a` as the linefeed code.

`crlf`: Outputs `0x0d`, `0x0a` as the linefeed code.

When specifying either `cr` or `crlf` by omitting a separator character, use a comma to separate them (e.g., `-W dat,, cr`).

`sup`

Specifies that trailing consecutive spaces are not to be output when column data of `CHAR`, `NCHAR`, or `MCHAR` type is output in DAT or extended DAT format.

For the `CHAR` or `MCHAR` type, the system compresses trailing single-byte spaces in a column with data that is less than the defined length. If the column data is all spaces, the system outputs one single-byte space.

For the `NCHAR` type, the system compresses trailing double-byte spaces in a column with data that is less than the defined length. If the column data is all spaces, the system outputs one double-byte space. The double-byte space depends on the character encoding in use. For details about the character encodings, see the *HiRDB Version 8 SQL Reference*.

For the output format when the `sup` option is specified, see Section *8.3.3 Format of database load utility input files*.

Criteria

Specifying the `sup` option has the following effects:

- The size of unload data file is reduced.
- If you specify the suppress option for a `CHAR`, `NCHAR`, or `MCHAR` column during table definition, the system creates input data for the database load utility in such a format that all trailing spaces are already compressed.

Notes

1. If the `sup` option is specified during an unload operation, the column data does not have an identical length in the unload data file. This point should be noted if the output data is to be sorted or edited with a UAP.
2. You can store multi-byte characters and a value other than character codes in a column of the `CHAR` or `MCHAR` type; however, if you specify the `sup` option, the command compresses spaces unconditionally even when a multi-byte character string contains a space character, or a non-character code has the same value as the space character. If this happens, you can pad any column shorter than the defined length with spaces by re-executing data loading on the same table. In this case, do not specify the `sup` option because if the data is referenced or displayed by another program, a coding error occurs on the last double-byte character.

`bin`

Specifies that the output data is in binary format.

`fixtext`

Specifies that data in the fixed-size data format is to be output.

Examples

1. When the padding character and `{cr|crlf}` are omitted

   ```
   -W fixtext
   ```

2. When the padding character is **Δ** and `{cr|crlf}` is omitted

   ```
   -W fixtext,Δ
   ```

3. When the padding character is **Δ** and `cr` is specified

   `-W fixtext,Δ,cr`

4. When the padding character is omitted and `crlf` is specified

   `-W fixtext,,crlf`

padding-character

Specifies the character to be used to fill out to the defined length when the column data or element data is the null value or when a variable-length character string is shorter than the defined length.

Rules

1. The padding character must be a 1-byte character.
2. When the padding character is omitted, the system assumes the space character (`0x20`). To use the space character as the padding character, omit *padding-character*.
3. The asterisk (`*`), double quotation mark (`"`), and underscore (_) must not be specified as the padding character.

`{cr|crlf}`

Specifies the linefeed code to be used.

`cr`:

Use the 1-byte linefeed code (`0x0a`).

`crlf`:

Use the 2-byte linefeed code (`0x0d`, `0x0a`).

(a) Criteria

Specify this option in the following cases:

- To migrate data from one table to another
- To output table data to an unload data file and then use the data with a UAP

(b) Rules

1. An unload data file obtained with the `-W` option specified cannot be used for reload processing by `pdrorg`.
2. If this option is specified, `hirdb` cannot be specified in the `-f` option.
3. When reorganizing a table in units of schemas, you cannot specify the `-W` option.
4. If the defined length of a table row exceeds 512 megabytes, data cannot be output in the DAT or fixed-size data format. In such a case, output the data in binary

format.

5. If you are transferring data from the UNIX version of HiRDB to the Windows version, or vice versa, output the data in the DAT format. If you need to convert the character encoding, do so first and then load data to the migration target.
6. If you specify the `-W` option for a table with LOB columns or columns of abstract data type with LOB attribute, either real data or the null value is output, depending on the `-j` option specification. Table 8-13 shows the relationship between options `-W` and `-j`.

Table 8-13: Relationship between options -W and -j

Column data type			**-W option**		
			bin	**dat or extdat**	**fixtext**
Predefined data type	`BINARY` type	`-j` option omitted	Real data	Real data	Not output
		`-j` option specified	Real data	Real data	Not output
	`BLOB` type	`-j` option omitted	Null value	Null value	Not output
		`-j` option specified	Real data	Null value	Not output
	Other	`-j` option omitted	Real data	Real data	Real data
		`-j` option specified	Real data	Real data	Real data
Abstract data type	`BLOB` attribute*	`-j` option omitted	Real data	Null value	Not output
		`-j` option specified	Real data	Null value	Not output
	Other	`-j` option omitted	Real data	Real data	Not output
		`-j` option specified	Real data	Real data	Not output

* This is the argument type of constructor parameter reverse creation function that is specified in the `unld_func` statement.

(8) -g

For a HiRDB/Parallel Server, this option specifies that table data stored in multiple servers is to be unloaded to a single unload data file at one server.

For a HiRDB/Single Server, this option specifies that an unload data file is to be created on the utility special unit.

Even when a table contains a LOB column, you can unload to a single LOB data unload file.

(a) Criteria

Specify this option in the following cases:

- Creating an unload data file for backup purposes
- Modifying the table partitioning conditions
- Outputting an unload data file to a utility special unit
- Allocating required space in another host with a HiRDB/Parallel Server when there is not enough space at the applicable host

(b) Notes

1. With a HiRDB/Parallel Server, the system assumes that the `-g` option is specified during table reorganization in units of schemas, even if this operand is omitted.
2. If the data unloaded in units of RDAREAs is reloaded in units of tables with the `-g` option specified, the data is deleted because the system treats other servers' RDAREAs as containing no data. In this case, do not specify the `-g` option.
3. If the data unloaded in units of RDAREAs is reloaded in units of RDAREAs with the `-g` option specified, you cannot modify the table partitioning conditions.

(9) -j

Specifies that a table containing LOB columns is to be unloaded.

You an also specify this option to unload a table containing an abstract data type with LOB attribute or reorganize LOB storage RDAREAs.

When this option is specified, the utility outputs both the LOB column structure base table and the LOB data to the unload data file during the unload operation. During the reload operation, the utility reloads both the LOB column structure base table and the LOB data. This feature enables a table containing LOB columns to be migrated to another system, or its partitioning condition to be changed. This depends on the unloading process being able to be executed as described in Section *8.4.3 Whether or not reloading is permitted when the table definitions of unload table and reload table do not match*.

(a) Rules

1. This option is ignored if specified for a table containing no LOB column.
2. The `-j` option is always assumed during reorganization in units of schemas.
3. If you specify the `-j` option during an unload operation, be sure to specify this option during a reload operation.
4. If the `-g` option is specified for unloading a table divided and stored in multiple servers in HiRDB/Parallel Servers, unloading is performed sequentially for each server at the expense of a reduction in parallel-processing performance.

5. If a table with `LOB` columns is to be unloaded using the `-W bin` option, the `-j` option sets the `LOB` columns as the columns to be unloaded.

(10) -u authorization-identifier

Specifies the authorization identifier of the user executing `pdrorg`.

For details about the default value, see *(b) Default value*, as follows.

When this option is specified, the system displays a message requesting entry of a password. If no password is required, enter null in response to the message.

The system uses the specified authorization identifier to connect to HiRDB and to check execution privileges.

(a) Criterion

Specify this option to use a different authorization identifier than the one specified in the `PDUSER` environment variable.

(b) Default value

When this option is omitted, the system assumes the following authorization identifier and password:

1. The system assumes the value of the `PDUSER` environment variable specified during execution of `pdrorg`. If you are executing the utility in the background with & attached by the shell, or in a remote shell environment where a response cannot be entered, be sure to specify the `PDUSER` environment variable. The following shows an example of specifying the `PDUSER` environment variable with the C shell:

   With password: setenv `PDUSER'"`*authorization-identifier*`"/"`*password*`"'`

   Without password: setenv `PDUSER'"`*authorization-identifier*`"'`

2. If the `PDUSER` environment variable is not specified, the system assumes the login window's user name. Enter the password when a message is displayed requesting password entry. If no password is required, enter null in response to the message.

(c) Rules

1. Do not specify this option if you are executing the utility in the background with & attached by the shell or in a remote shell environment where a response cannot be entered.

2. The system treats an authorization identifier enclosed in double quotation marks (`"`) as case sensitive; otherwise, the system treats it as in all uppercase letters. If you use the Bourne shell (`sh`), C shell (`csh`), or Korn shell (`ksh`), you must enclose the authorization identifier in single quotation marks (`'`).

(11) -n [batch-input/ output-local-buffer-sectors-count],[div],[random-access-local-buffer-sectors-count]

Specifies that a local buffer is to be used for reading data from the database during an unload operation or for writing data to the database during a reload operation. When this option is specified, the number of input/output operations can be reduced by the number of batch input/output operations because the system uses the local buffer to access databases.

When this option is omitted, the system uses the global buffer to access one page at a time during an input/output operation.

batch-input/output-local-buffer-sectors-count ~ <unsigned integer> ((2-4096))

Specifies the number of batch input/output local buffer sectors. The batch input/ output local buffer is used for data pages.

We recommend that you specify a value in the range 16-32 for the number of batch input/output local buffer sectors. The guideline is 64 kilobytes/page size.

`div`

Specify `div` when all the conditions listed below are satisfied. If all these conditions are satisfied but `div` is not specified, the number of input/output operations may increase, thereby affecting performance adversely.

- The partitioning conditions during the unload operation differ from those during the reload operation.
- There are multiple RDAREAs for a table in the same server.

When `div` is specified, the required memory size increases because the system allocates as many buffer sectors as there are table partitions in the server.

random-access-local-buffer-sectors-count ~ <unsigned integer> ((4-125000))

Specifies the number of random access local buffer sectors. The random access local buffer is used for index pages.

Hitachi recommends that you change the combination of the number of batch input/ output local buffer sectors, `div`, and the number of random access local buffer sectors according to the table definition. Table 8-14 shows the recommended `-n` option value.

Table 8-14: Recommended -n option specification (pdrorg)

Condition	Table type	Column definition	Table partitioning in server	
			Yes	No
During unloading	FIX table	●	`-n` *x*, `div`	`-n` *x*

Condition	Table type	Column definition	Table partitioning in server	
			Yes	No
	Non-FIX table	●	`-n`, , *y*	
During reloading	FIX table	●	`-n` *x*, `div`	`-n` *x*
	Non-FIX table	A variable-length character string with a column length of 256 bytes or more or a BINARY column is defined.	`-n`, , *y*	
		An abstract data type column is defined.	`-n` *x*, `div`, *y*	`-n` *x*, , *y*
		A repetition column is defined.		
		Other	`-n` *x*, `div`	`-n` *x*

Legend:

x: Number of batch output local buffer sectors

y: Number of random access local buffer sectors

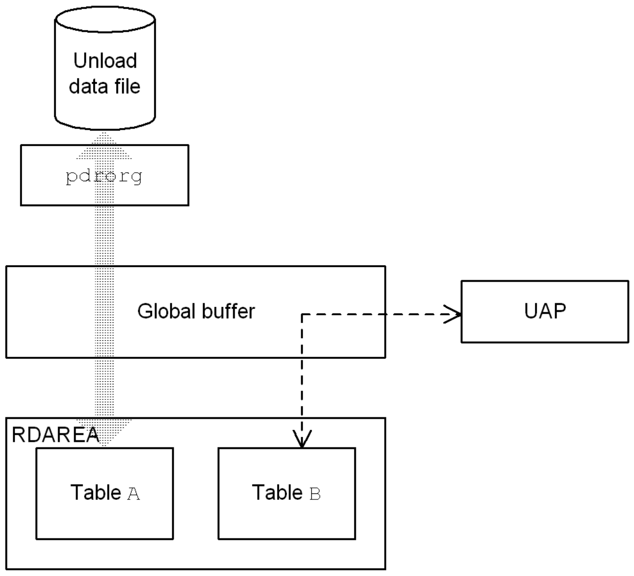
●: Not applicable

(a) Buffer used by pdrorg

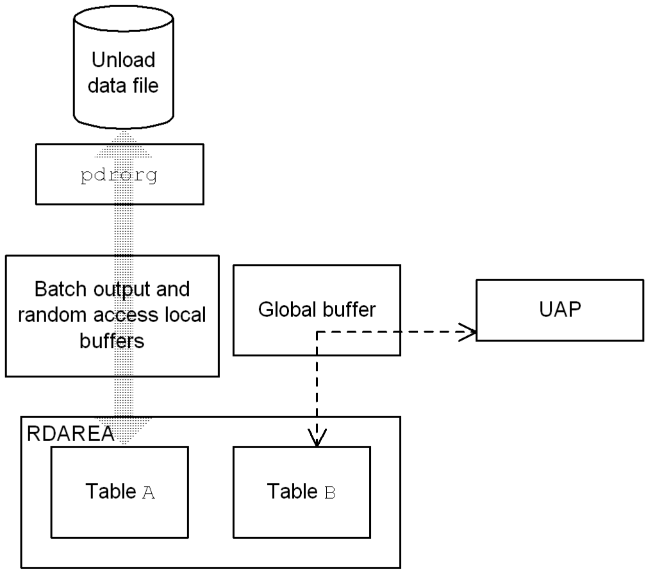
1. When the `-n` option is omitted, the `pdrorg` command uses the global buffer. In such a case, transaction performance is degraded for a process that uses the global buffer because a large amount of global buffer space is needed for reorganization. When the `-n` option is specified, there is no such buffer contention. Figure 8-17 shows the relationship between `pdrorg` and the buffer:

Figure 8-17: Relationship between pdrorg and the buffer

● When using only a global buffer



● When using both local and global buffers



Explanation:

When only the global buffer is used (`-n` option is omitted), contention occurs on the buffer between `pdrorg` and UAPs.

When both a local buffer and a global buffer are used (`-n` option is specified), no contention occurs on the buffer between `pdrorg` and UAPs. Note that during reorganization of a LOB column, the system uses the global buffer even if the `-n` option is specified.

2. If you specify only the number of batch input/output local buffer sectors in the `-n` option, the system uses only one batch input/output local buffer sector for a single RDAREA. This may result in buffer contention because the system uses only one batch input/output local buffer sector even if there are multiple RDAREAs. If buffer contention occurs, the number of input/output operations increases, affecting performance adversely. In such a case, specify `div`. When `div` is specified, the system allocates as many batch input/output local buffer sectors as there are RDAREAs (one batch input/output local buffer sector for each RDAREA), thereby avoiding buffer contention.

3. Even if a batch input/output local buffer is specified, the system may use the global buffer depending on conditions. Tables 8-15 and 8-16 describe the relationships between the conditions and the buffer.

Table 8-15: Relationships between conditions and the buffer to be used (during an unload operation by pdrorg)

Condition			**Random access local buffer**				
			Not specified		**Specified**		
			Global buffer	**Batch input/ output local buffer**	**Global buffer**	**Batch input/ output local buffer**	**Random access local buffer**
Data page	RDAREA storing LOB column structure base table	FIX table	—	Y	—	Y	—
		Non-FIX table	Y	—	Y	—	—
	RDAREA storing LOB column		Y	—	Y	—	—
	RDAREA storing LOB attributes		Y	—	Y	—	—
Index page	When searching key values from index (when `-b` is specified)		Y	—	—	—	Y

Condition	Random access local buffer				
	Not specified		Specified		
	Global buffer	Batch input/ output local buffer	Global buffer	Batch input/ output local buffer	Random access local buffer
Directory page	Y	—	Y	—	—

Legend:

Y: Used

— : Not used

Table 8-16: Relationships between conditions and the buffer to be used (during a reload operation by pdrorg)

Condition				Random access local buffer				
				Not specified		Specified		
				Global buffer	Batch input/ output local buffer	Global buffer	Batch input/ output local buffer	Random access local buffer
Data page	RDAREA storing LOB column structure base table	FIX table		—	Y	—	Y	—
		Non-FIX table	When a variable-length character string or BINARY string with column length greater than 256 bytes is defined	Y	—	Y	—	—
			When a column of an abstract data type or a repetition column is defined and a row of data is too large to fit on one page	Y	—	Y	—	—
			Other	—	Y	—	Y	—

Condition		Random access local buffer				
		Not specified		Specified		
		Global buffer	Batch input/ output local buffer	Global buffer	Batch input/ output local buffer	Random access local buffer
	RDAREA storing a LOB column	Y	—	Y	—	—
	RDAREA storing LOB attributes	Y	—	Y	—	—
Index page	When searching key values from table (when `-i c` or `-i n` is specified)	—	Y	—	Y	—
	When creating index concurrently (when `-i s` is specified)	Y	—	—	—	Y
	When executing batch index creation (when `-i c` is specified)	Y	—	—	—	Y
Directory page		Y	—	Y	—	—

Legend:

Y: Used

— : Not used

(b) Rules

1. For rebalancing a table, performance may reduce considerably if the size of the global buffer is insufficient. If none of the columns exceeds 256 bytes in defined length, specify the number of batch output pages.

   If the local buffer specification does not take effect for some reason, such as the existence of a column with a defined length of 256 bytes or greater, allocate at least the following number of global buffer sectors:

   Number of buffer sectors required per RDAREA

   = 1024 $\div$ (number of table storage RDAREAs) $\times$ 2 + 3

2. For rebalancing a table using the FIX hash partitioning method, the system allocates the buffer for the number of pages specified for each hash group. Therefore, more memory may be required than for a non-partitioned table or a table partitioned by other methods.

(12) -f unload-data-file-type-or-LOB-data-unload-file-type

Specifies the type of unload data file or LOB data unload file.

When this option is omitted, the utility assumes a regular file or streaming tape device as the unload data file.

`easymt`

EasyMT is used as the unload data file or LOB data unload file.

For a HiRDB/Parallel Server, if you have specified `hirdb`

A HiRDB file is used as the unload data file or LOB data unload file.

(13) -b unload-sequence

Specifies the order in which data is to be unloaded.

When this option is omitted, if the cluster key is defined for the table, the utility unloads the table data in the order of the cluster key values; otherwise, the utility unloads the table in the order that its data is stored.

You cannot specify this option for an index, plug-in index, or index for a repetition column with an exception value specification.

(a) Criteria

To unload a table with a cluster key defined, omit this option. If a cluster key has not been defined, but the table is to be unloaded and then a cluster key is to be defined for it before reloading, you must unload the table in the order of index values that is the same as the order of cluster key values (`ASC` or `DESC`).

Specify one of the following options as the order of data to be unloaded:

`cluster`

Unload data in the order of the cluster key values

`index,`*index-identifier*`[,desc]`

Unload data in the order of the index values.

`desc` unloads in the descending order of index values. Note that `desc` is applicable only to a single-column index.

`primary[,,desc]`

Unload data in the order of primary key values.

`desc` unloads data in the descending order of index values. Note that `desc` is applicable only to a single-column index.

`physical`

Unload data in the order it is stored.

Rules

1. The system treats an index identifier enclosed in double quotation marks (`"`)

as case sensitive; otherwise, the system treats it as in all uppercase letters. If an index identifier contains a space, enclose the index identifier in double quotation marks ("). If using the Bourne shell (`sh`), C shell (`csh`), or Korn shell (`ksh`), you must enclose the index identifier in single quotation marks (').

2. The `desc` option is applicable only to single-column indexes.
3. When this option is specified, data is output to the unload data file in units of table storage RDAREAs in the order in the specified index, but it is not in the order in the index in the table as a whole. If you need an unload data file in which data is sorted in the order in the index (such as when you output data in DAT format by specifying the `-W` option), you must create an unload data file in units of RDAREAs for a row-partitioned table.
4. You cannot specify the `-b` option for reorganization in units of schemas.
5. When you are unloading LOB columns in the order of index values (by specifying `index` in the `-b` option), the index must be available. As in the following cases, if no index has been created, you need to create the index first and then unload the LOB columns:

- `n` specified in the `-i` option
- Reloading to a non-partitioned table in units of RDAREAs for which a non-partitioning key index is defined

(14) -s

When reorganizing a table with the suppress option specified for a `CHAR`, `NCHAR`, or `MCHAR` column, specify this option to remove all trailing consecutive spaces from columns during the unload operation.

(a) Effects of this option

By specifying this option, you can reduce processing time and the size of unload data file.

The following table shows the cases for which specifying the `-S` option can reduce processing time and the size of unload data file:

Status of unloaded table		Status of reloaded table			
		Column without suppress option		Column with suppress option	
		Condition A	Condition B	Condition A	Condition B
Column with suppress option	Condition A	M	M	Y	M
	Condition B	—	—	—	—

Condition A: Applicable column has no index defined and is neither key range-partitioned nor hash-partitioned.

Condition B: Applicable column has an index defined or is key range-partitioned or hash-partitioned.

Y: The unload and reload processing time is reduced and the size of the unload data file is also reduced.

M: The unload processing time is reduced and the size of the unload data file is also reduced, but the reload processing time is slower than the case indicated as .

—: There is no effect on the reduction of processing time or size of the unload data file.

(b) Notes

1. For the following columns, specifying the `-S` option cannot reduce the processing time or the size of the unload data file, even if the suppression option is specified for the corresponding table:
   - Columns defining index
   - Columns partitioned by key range partitioning method
   - Columns partitioned by hash partitioning method
2. If the definitions of these columns have changed during the reload operation, it takes more processing time than when the definitions have not changed.
3. If the `-W` option is specified, the `-s` option, if specified, is ignored.

(15) -o

Specify this option to automatically delete the index information files after index creation processing is completed successfully.

(a) Criteria

A plug-in index in particular requires a large index information file. Such index information files require a large amount of disk space, if retained. You can specify this option to make sure that the index information files are always deleted after index creation processing.

(b) Notes

The following table shows the relationship between index information files and the `-o` option:

-k option	Contents of index information file	-o option	
		Specified	Omitted
`rorg`, `reld`	Index information file specified in the `index` statement	Y*	—
	Index information file allocated by HiRDB	Y*	Y*
`ixmk`	Index information file created by `pdload` or `pdrorg`	Y	—
	Index information file created by the plug-in index delayed batch creation facility	Y	Y
`ixrc`, `ixor`	Index information file allocated by HiRDB with the `idxname` specified	Y	Y
	Index information file specified in the `index` statement	Y	—

Y: Deleted.

—: Not deleted.

* For a table partitioned within the server, the index information file is not deleted if its non-partitioning key index is not partitioned within the server during reorganization in units of RDAREAs.

(16) -m progress-message-output-interval

~ <unsigned integer> ((1-1000)) <<10>>

Specifies in units of 10,000 rows the interval at which execution status messages are to be output.

(a) Criterion

Specify this option to change the default value of 10,000 rows.

(b) Notes

1. When the batch index creation mode is specified or batch index creation is underway during a table reload operation, the utility ignores this option, in which case messages are output at the time index creation is started and terminated.
2. This option is ignored if `ixrc` is specified in the `-k` option. However, this option take effect for the processing of plug-in indexes.

(17) -X response-monitoring-time-for-server-to-server-communication

~ <unsigned integer> ((1 ~ 65535)) <<300>>

If an error, such as a communication error, occurs at the server where the command was executed, the command may stop responding and the application may stop. To help you detect errors, `pdrorg` enables you to monitor a response time during communication for dictionary manipulation that is performed by the command.

In the `-X` option, set the response monitoring time (in seconds) during dictionary manipulation. If the execution time during dictionary manipulation exceeds the value set in the `-X` option, `pdrorg` treats it as a dictionary access error and cancels processing with return code `8`.

Criteria

- If you want to detect an error in fewer than 300 seconds in the event of a no-response from the server due to a communication error or unit down, specify a value less than 300 in the `-X` option.
- If the system switchover facility is used, the command may keep waiting for a response even though system switchover has been completed. In such a case, you can terminate the command immediately by reducing the monitoring time.
- The specified monitoring time may result in a timeout if a response from the dictionary is delayed and if the utility's preprocessing is not completed within 300 seconds, which is the default value for the `-X` option. This can happen when many applications and utilities are executing concurrently. In such an environment, specify a value greater than 300 in the `-X` option.

(18) -q generation-number

~ <unsigned integer> ((0-10))

Specifies the generation number of the target RDAREA when the inner replica facility is used.

Specify the generation number as follows:

`0`: When the original RDAREA is to be processed

`1` to `10`: When the replica RDAREA with the specified generation number is to be processed

(a) Criteria

Specify this option when you are using the inner replica facility and processing an RDAREA that is not the current RDAREA.

(b) Rules

1. When you are not using the inner replica facility, this option is not available.
2. When this option is omitted, the current RDAREA becomes subject to processing.
3. If you are processing a replica RDAREA, use the `-r` option to specify the name of the original RDAREA and the `-q` option to specify the generation number that is to be subject to processing.

4. `pdrorg` checks the generation of the target RDAREA for errors. If the following checking results in an error, `pdrorg` displays a message and terminates with return code `8`:

   Checking for an inconsistent generation number for a row-partitioned table:

   When a row-partitioned table is to be processed, `pdrorg` checks the generations of target RDAREAs.

   When the `-q` option is specified, `pdrorg` checks each target for the specified generation of RDAREA. When the `-q` option is omitted, `pdrorg` checks the current RDAREA at each target for any inconsistency in the generation number.

   Checking for any inconsistent generation number for the RDAREAs storing the table and index:

   `pdrorg` checks the generations of the RDAREAs storing the table and index.

   When the `-q` option is specified, `pdrorg` checks the target RDAREA for the specified generation. When the `-q` option is omitted, `pdrorg` checks all the target current RDAREAs to determine whether or not their generations are the same.

   Checking for any inconsistent replica status for the target RDAREA:

   When the `-q` option is specified, `pdrorg` checks the target RDAREAs to determine whether any non-current RDAREA is mixed in the current RDAREAs.

5. The following table describes whether or not each file is available depending on the use of inner replica facility:

File output condition			**Availability of file**			
			Inner replica facility not used	**Inner replica facility used**		
				Original RDAREA	**Replica RDAREA**	
					Same generation[*1]	**Different generation**
Index information file	Inner replica facility not used		Y	Y	N	N
	Inner replica facility used	Original RDAREA	Y	Y	N	N
		Replica RDAREA	N	N	Y	N

File output condition			Availability of file			
			Inner replica facility not used	Inner replica facility used		
				Original RDAREA	Replica RDAREA	
					Same generation*1	Different generation
Unload data file	Inner replica facility not used		Y	Y	Y*2	Y*2
	Inner replica facility used	Original RDAREA	Y	Y	Y*2	Y*2
		Replica RDAREA	Y*2	Y*2	Y	Y*2

Legend:

Y: File can be used.

N: File cannot be used.

*1 This is the case where the generation information during file output is the same as the generation information specified in the `-q` option. When the `-q` option is omitted, this is the case where the generation information during file output is the same as the generation information for the current RDAREA.

*2 The file can be used when the following conditions are satisfied:

- Conditions for reloading in *8.4.3 Whether or not reloading is permitted when the table definitions of unload table and reload table do not match* are satisfied.
- There is a LOB column or a column of an abstract data type with the LOB attribute and the `-j` option is specified.

(19) control-information-filename

~ <pathname>

Specifies the name of the control information file that contains `pdrorg`'s control statements.

You can specify the following control statements in the control information file. For details about each control statement, see Sections *8.9.3* to *8.9.18*.

Control statement	pdrorg's function						
	Table reorg	Table unldg	Table reldg	Data dict table reorg	Batch IX cr	IX recr	IX reorg
`mtguide` statement `emtdef` statement	O	O	O	O	—	—	—
`unload` statement	R	R	R	R	—	—	—
`index` statement	O	—	O	—	R	R[6]	R[6]
`idxname` statement	—	—	—	—	—	R[6]	R[6]
`idxwork` statement	O[1]	—	O[1]	—	—	O	O
`sort` statement	O[2]	—	O[2]	—	O	O	—
`lobunld` statement	O	O[3]	O	—	—	—	—
`unlduoc` statement	O	O	—	E	E	E	E
`tblname` statement	—	—	O	—	—	—	—
`array` statement	—	O[4]	—	—	—	—	—
`unld_func` statement	O[5]	O[8]	—	—	—	—	—
`reld_func` statement	O[5]	—	O	—	—	—	—
`constraint` statement	O	—	O	—	—	—	—
`option` statement	O	O[4, 7]	O	O[7]	O[7]	O[7]	O[7]
`report` statement	O	O	O	O	O	O	O
`blobtovarchar` statement	—	O[4]	—	—	—	—	—
`fixtext_option` statement	—	O[4]	—	—	—	—	—

Legend:

Table reorg: Table reorganization

Table unldg:Table unloading

Table reldg: Table reloading

Data dict table reorg: Data dictionary table reorganization

Batch IX cr: Batch index creation

IX recr: Index re-creation

IX reorg: Index reorganization

R: Required.

O: Optional.

—: Ignored, if specified.

E: Results in an error, if specified.

1 If `-i s` is specified, this statement cannot be specified.

2 If `-i n` or `-i s` is specified, this statement cannot be specified.

3 If the `-W` option is specified, this statement cannot be specified.

4 This statement can be specified only when the `-W` option is specified.

5 Both `unld_fund` and `reld_func` statements must be specified.

6 Specify either the `index` or `idxname` statement.

7 This statement cannot be specified for a table reorganization with the synchronization point specification.

8 This statement is required if a table with columns of abstract data type is to be unloaded with `-W bin` specified.

(a) Relationship between control statements and options

Table containing a LOB column or abstract data type column provided by a plug-in (LOB attribute)

When a table containing a LOB column is to be reorganized, the unit of reorganization depends on the specification of the `unload` and `lobunld` statements. The following table shows the relationship between the unit of reorganization for a table containing a LOB column and the `-j` option in the `unload` and `lobunld` statements:

Reorganization unit	unload statement		lobunld statement	
	-j specified	-j omitted	-j specified	-j omitted
LOB column structure base table and LOB data	Y	Y	N	Y
LOB column structure base table	N	Y	N	—
LOB data	N	—	N	Y

Y: Can be specified.

N: Cannot be specified.

— : Not required.

Index re-creation or reorganization

When an index is re-created or reorganized, you can use the `idxname` statement to process the index in units of indexes or use the `index` statement to process the index in units of index storage RDAREAs. The following table shows the relationship between the index re-creation or reorganization and the `index`, `idxname`, and `idxwork` statements:

Index re-creation or reorganization unit	index statement	idxname statement	idxwork statement	sort statement	
				-k ixrc	-k ixor
Indexes	N	R	O	O	—
Index storage RDAREAs	R	N	—	O	—

R: Required.

O: Optional.

N: Cannot be specified.

— : Ignored, if specified.

(b) Files and directories specified in the control statements

The following rules apply to files and directories specified in the control statements:

1. You must grant access privileges to the HiRDB administrator in advance. In some cases when control statements or operands are omitted, the system assumes that the applicable file is to be created in the `/tmp` or `/usr/tmp` directory; for this reason, you must also grant access privileges to the `/tmp` or `/usr/tmp` directory.
2. For the names of the unload data files, index information files, and EasyMT's MT attribute files, assign unique names, regardless of their hosts.

8.9.3 mtguide and emtdef statements (specification of MT information)

To use EasyMT as an unload data file or LOB data unload file, specify the mtguide and emtdef statements.

The mtguide and emtdef statements must precede the unload or lobunld statement.

(1) Format

```
mtguide{use|nouse}
emtdef MT-attribute-definition-filename
```

(2) Explanation

(a) mtguide{use|nouse}

When magnetic tape is used for the unload data file or LOB data unload file, this option specifies whether MTguide is to be used for the magnetic tape mounting operation.

`use`

MTguide is to be used.

`nouse`

MTguide is not to be used.

(b) emtdef MT-attribute-definition-filename

~ <path name>

Specifies the name of the file that defines EasyMT's MT attribute. The following items specified in this file take effect on EasyMT processing:

- bufno: Number of I/O buffer sectors
- magazin: MT device allocation pattern
- job: Application name
- preserve: Retention days
- expire: Expiration date

However, if the same item is specified in the unload statement or lobunld statement, the unload statement or lobunld statement specification takes precedence.

HiRDB/Single Server

Create the MT attribute definition file at the server machine that contains the single server.

HiRDB/Parallel Server

Create the MT attribute definition file at the server machine where the system manager is located.

(3) Notes

To use EasyMT, you need to initialize the MT to be used for the unload operation to the EasyMT-labeled volume format beforehand using EasyMT's EmtInit function or the emtinit command.

To use a multi-volume MT, you need to initialize all volumes.

The following shows the methods for specifying the initialization parameter:

Specification method	Value
Specifying in the volume attribute definition table	`EMTVOL_EL`
Specifying in the MT attribute definition file	`voltype=EL`

8.9.4 unload statement (specification of unload data file)

The `unload` statement specifies information about an unload data file.

Criterion

If you specify `-k rorg`, `-k unld`, or `-k reld`, be sure to specify the `unload` statement.

Rules

You can specify the following number of `unload` statements:

HiRDB/Single Server

Specify only one `unload` statement, regardless of the reorganization units (tables or RDAREAs).

HiRDB/Parallel Server

When reorganizing a row-partitioned table in units of tables, specify as many `unload` statements as there are servers that contain the table storage RDAREAs.

When reorganizing a row-partitioned table in units of RDAREAs or reorganizing a non-partitioned table, specify only one `unload` statement. Also, when unloading a table to a single unload data file specifying the `-g` option, specify only one `unload` statement.

(1) Format

```
unload {[{server-name:|host-name:}]
              unload-data-file-name[,unload-data-file-name]...
               |(uoc)}
               [{EasyMT-information|HiRDB-file-information}]
               [uoc-information]
```

(2) Explanation

(a) server-name

~ <identifier> ((1-8))

Specifies the name of the server used to create the unload data file.

HiRDB/Single Server

Do not specify this option for a HiRDB/Single Server.

HiRDB/Parallel Server

When reorganizing a row-partitioned table in units of tables, specify the name of the server used to create the unload data file.

When reorganizing a row-partitioned table in units of RDAREAs or reorganizing a non-partitioned table, there is no need to specify this option.

When unloading a single unload data file specifying the `-g` option, specify the name of the server used to create the unload data file.

(b) host-name

~ <identifier> ((1-32))

Specifies the name of the host used to create the unload data file.

HiRDB/Single Server

When creating the unload data file on a utility special unit, specify the name of that host.

When this option is omitted, the system creates the unload data file on the unit where the single server is located.

HiRDB/Parallel Server

Do not specify this option for a HiRDB/Parallel Server.

(c) unload-data-filename

~ <pathname>

Specifies the absolute path name of the unload data file.

You can specify the following files as unload data files:

- Regular file

  You can create these files in a file system area provided by the operating system, without any preliminary steps.

- Character special file

  Create these files in a HiRDB file system area for utilities that was created by the `pdfmkfs` command.

If you have specified the type of unload data file in the `-f` option, specify the unload data file names as follows.

When the unload data file is EasyMT:

You can specify a maximum of two unload data files.

If using MTguide, you can specify a device symbolic name or device group name managed by MTguide.

When the unload data file is a HiRDB file:

- Express the name of the unload data file as 1 to 167 characters.
- To use a HiRDB file, the HiRDB file system area must have already been created by the `pdfmkfs` command, with `UTL` specified as the usage purpose in the `-k` option of the `pdfmkfs` command.
- An error will result if the specified HiRDB file is in a HiRDB file system area that was created by the `pdfmkfs` command with an option other than `-k UTL` specified.
- If the specified HiRDB file name is not found in the HiRDB file system area, `pdrorg` creates a new HiRDB file. If you specify the name of an existing HiRDB file, the utility overwrites that file.

Note about unload data files

1. If there is too much table data to fit in one unload data file, you can specify multiple files.
2. When specifying multiple unload data files, note the following:
   - You cannot use an NFS file.
   - You cannot use a direct tape unit without using EasyMT.

   The file attributes must be the same as shown as follows:
   - EasyMT

     You need to initialize all volumes as EMTVOL_EL (EasyMT-labeled volumes).
   - HiRDB files

     You cannot mix regular files and character special files. Use either regular files only or character special files only.

(d) (uoc)

Specify this option when you use a UOC to reorganize the database or to create a UOC data file (when the unload data file is not output by `pdrorg`).

`(uoc)` is applicable only to `-k unld`. If `(uoc)` is specified, neither EasyMT information nor HiRDB file information can be specified.

(e) EasyMT-information

```
[file=filename]
[{,vol=volume-name|,vol=(volume-name[,volume-name]...)}]
[,bufno=buffer-sectors-count]
[,fileno={file-sequence-number|ADD}]
```

Specify this operand if the unload data file is an EasyMT file.

By specifying a file name and volume name, you can check to see if the actual unload data file and volume have the specified names. If you do not want to check the file or volume name, omit this operand.

If you are omitting a file name, do not specify a comma before the first specified item.

`file=`*filename* ~ <alphanumerics> ((1-17))

Specifies the name of the file.

`{,vol=`*volume-name*`|,vol=(`*volume-name*`[,`*volume-name*`]...)}`

~ <alphanumerics> ((1-6))

Specifies the names of the volumes containing the file. You can specify up to 255 volume names, in as much as they fit in one line.

`bufno=`*buffer-sectors-count* ~ <unsigned integer> ((1-256)) <<10>>

Specifies the number of buffer sectors to be used by EasyMT.

`fileno={`*file-sequence-number*`|ADD}`

Specifies the position of the unload data file on the magnetic tape.

file-sequence-number ~ <unsigned integer> ((1-199)) <<1>>

If you have specified -k rorg or -k unld, the utility creates a file at the specified location and deletes any file following the specified location.

If you have specified -k reld, the utility reads the file at the specified location (if fileno is omitted, the utility reads the next file).

`ADD`

If you have specified -k rorg or -k unld, the utility creates the file at the end of the volume.

If you have specified -k reld, you cannot specify ADD.

(f) HiRDB-file-information

```
[init={initial-allocation-size
    |(initial-allocation-size[,initial-allocation-size]...)}]
[,incr=secondary-allocation-size
    |(secondary-allocation-size[,secondary-allocation-size]...)}
```

Specify this operand if the unload data file is a HiRDB file. Be sure to specify the HiRDB file information on a single line.

If you omit `init`, do not specify a comma (`,`) before `incr`.

`init={`*initial-allocation-size*| `(`*initial-allocation-size*`[,`*initial-allocation-size*`]...)}` ~ <unsigned integer> ((1-1048574)) <<100>>

Specify in MB the size of the area to be allocated to write to the HiRDB file.

Estimation formula:

For details about determining the initial allocation size, see the section in the *HiRDB Version 8 Installation and Design Guide* that describes the file size during execution of the database reorganization utility (`pdrorg`) to obtain the size (in bytes) of the unload data file, then use the following formula:

Initial allocation size = ↑(size of unload data file in bytes + 1024) ÷ (1024 × 1024)↑ (MB)

Rules

1. The initial allocation size must be smaller than the value specified in the `-n` option when the HiRDB file system area was created with the `pdfmkfs` command (because the system requires an area for management purposes).
2. Be sure to specify a nonzero value in the `-e` option of the `pdfmkfs` command. Otherwise, the secondary allocation is not possible.
3. Use the `pdfstatfs` command to determine the initial allocation size permitted.
4. Specify as many allocation sizes as there are files specified in the `unload` statement. The system resolves a mismatch between the number of initial allocation sizes and the number of file names as follows:

Condition	Resolution
Number of files < number of initial allocation specifications	As many initial allocation specifications are valid as there are files, and subsequent initial allocation specifications are ignored.

Condition	Resolution
Number of files = number of initial allocation specifications	Specification for each file is valid.
Number of files > number of initial allocation specifications	The last initial allocation specification is assigned to all the remaining files.

`incr=`*secondary-allocation-size*|
`(`*secondary-allocation-size*`[,`*secondary-allocation-size*`]...)}` ~ <unsigned integer> ((1-1048574)) <<10>>

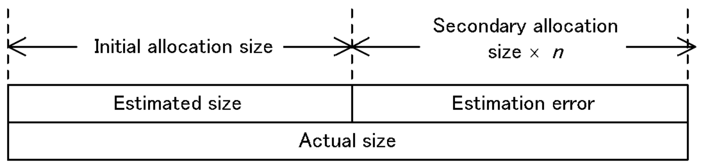
Specifies the size of the extension area when the initial allocation size is not sufficient during a write operation on the HiRDB file.

The utility does not expand the HiRDB file in the following cases:

- Attempt to expand the corresponding HiRDB file system area will exceed the specified number of extensions.
- Attempt to expand the corresponding HiRDB file will exceed the maximum number of extensions for individual HiRDB files (23 times).

If the actual size exceeds the estimated size, specify the secondary allocation size in such a manner that neither of the previous attempts is made. Figure 8-18 shows limitations to the secondary allocation size. Specify the value of *n* so that the above two attempts are avoided.

Figure 8-18: Limitations to the secondary allocation size (unload statement)



When creating a new HiRDB file system area, you can specify the number of secondary allocations in the `-e` option of the `pdfmkfs` command. For an existing HiRDB file system area, check the value of `available expand count` in the `pdfstatfs` command output result.

(g) uoc-information

`[uoc_lib=`*library-name*`[param='`*user-parameters*`']`

Specify UOC information when you use a UOC to reorganize the database. When specifying the `unlduoc` statement, make sure that UOC information is also specified (if it is omitted, an error results).

`uoc_lib=`*library-name*

～ <path name>

Specifies the absolute path name of the shared library that contains the UOC.

`param='`*user-parameters*`'`

～ <character string> ((1-1023))

Specifies user parameters that are to be passed to the UOC (because the user cannot directly pass parameters to a UOC by methods such as the command line).

The information specified in this option is passed as is to the UOC via the UOC interface area. The character string specified as the user parameters cannot contain any spaces or tabs.

8.9.5 index statement (specification of index information file information)

The `index` statement specifies information about index information files.

Criteria

- `-k rorg` or `-k reld` specified

  If possible, always specify the `index` statement in order to avoid a shortage of space in the `/tmp` directory.

  If there are many indexes or index storage RDAREAs, you should specify the `idxwork` statement.

- `-k ixmk` statement

  Be sure to specify the `index` statement.

- `-k ixrc` or `-k ixor` specified

  Specify the `index` statement to re-create an index in units of index storage RDAREAs or to reorganize an index (to process an index in units of indexes, specify the `idxname` statement).

Rules

1. Specify one `index` statement for each index storage RDAREA.
2. There is no need to specify an `index` statement when you are reorganizing a table with no index defined or using the index update mode.
3. If the `index` statement is omitted and the `idxwork` statement is also omitted, the utility creates index information files in the `/tmp` directory of the server that contains the index.
4. If both the `index` and `idxwork` statements are specified, the `index` statement takes effect.

(1) Format

```
index index-identifier[RDAREA-name] index-information-filename
```

(2) Explanation

(a) index identifier

Specify the identifier of the index.

The system treats an index identifier enclosed in double quotation marks (") as case sensitive; otherwise, the system treats it as all uppercase letters. Enclose an index identifier in double quotation marks if it contains a space.

(b) RDAREA-name

~ <identifier> ((1-30))

For a row-partitioned table, specify the name of the RDAREA containing a table partition.

If you are specifying a replica RDAREA, specify its original RDAREA name in this option and then specify the target generation number in the `-q` option.

The system treats an RDAREA name enclosed in double quotation marks (") as case sensitive; otherwise, the system treats it as all uppercase letters. Enclose an RDAREA name in double quotation marks if it contains a space.

(c) index-information-filename

~ <pathname>

Specify the absolute path name of the index information file to which index information is to be output.

If you are re-creating an index using the index information file output by the delayed batch index creation facility, you can use the name of a HiRDB file for the index information file (express "*HiRDB-file-system-area-name*/*HiRDB-file*" as 1 to 167 characters). Otherwise, specify the name of a regular file.

8.9.6 idxname statement (specification of index information)

The `idxname` statement specifies information about an index to be re-created or reorganized.

Criterion

Specify the `idxname` statement to re-create or reorganize an index in units of indexes (to process an index in units of index storage RDAREAs, specify the `index` statement).

Rules

1. You can specify as many `idxname` statements as there are indexes. However, if `name=*` is specified, then only one `idxname` statement can be specified, in which case specifying more than one `idxname` statement results in an error.
2. If the `idxname` statement is specified, the utility processes all index storage RDAREAs that constitute the index. This means that multiple `index` statements may be replaced by only one `idxname` statement.
3. When the `idxname` statement is specified, the utility uses only one directory to output index information files, requiring a large space for the directory. If the `idxwork` statement is omitted and there is not much space available in the `/tmp` directory, a space shortage may occur. In such a case, specify a directory with sufficient free space in the `idxwork` statement.

(1) Format

```
idxname name=index-identifier [server=server-name[,server-name]...]
```

(2) Explanation

(a) name=index-identifier

Specifies the identifier of the index.

You can specify an asterisk (`*`) for the index identifier, in which case the utility re-creates all indexes defined for the table.

The system treats an index identifier enclosed in double quotation marks (`"`) as case sensitive; otherwise, the system treats it as all uppercase letters. Enclose an index identifier in double quotation marks if it contains a space.

(b) server=server-name

~ <identifier> ((1-8))

For a HiRDB/Parallel Server, this option specifies a server name.

If you specify a server name, the utility processes only those indexes at the specified server. If you omit this option, the utility processes all indexes at all servers.

(3) Relationship between operands and the execution environment

The following table shows the relationship among the `name` operand, `server` operand, and execution environment:

name operand	server operand	Execution environment	
		HiRDB/Single Server	HiRDB/Parallel Server
index-identifier	*server-name*	Not applicable.	The utility processes all index storage RDAREAs that constitute the specified index at the specified server.
	Omitted	The utility processes all index storage RDAREAs that constitute the index.	The utility processes all index storage RDAREAs that constitute the specified index at all servers.
*	*server-name*	Not applicable.	The utility processes all index storage RDAREAs that constitute all indexes defined for the table at the specified server.
	Omitted	The utility processes all index storage RDAREAs that constitute the index defined for the table.	The utility processes all index storage RDAREAs that constitute all indexes defined for the table at all servers.

8.9.7 idxwork statement (specification of directory for index information file)

The `idxwork` statement specifies the name of a directory in which index information files are to be created automatically when the `index` statement is omitted.

Criterion

Always specify the `idxwork` statement, if possible, in order to avoid a shortage of space in the `/tmp` directory.

Rules

1. If both `index` and `idxwork` statements are omitted, the utility creates index information files in the `/tmp` directory directory at the server that contains the index storage RDAREAs.
2. You can specify as many `idxwork` statements as follows:

   HiRDB/Single Server

   Specify only one `idxwork` statement.

   HiRDB/Parallel Server

   For an index of a partitioned table, specify as many `idxwork` statements as there are servers that contain the table partitions. For a non-partitioned table or when reorganizing a partitioned table in units of RDAREAs, specify only one `idxwork` statement.
3. If both `idxwork` and `index` statements are specified, the `index` statement takes effect, in which case the `idxwork` statement is ignored.

(1) Format

```
idxwork [server-name] directory-name
```

(2) Explanation

(a) server-name

~ <identifier> ((1-8))

Specifies the name of the server used to create index information files.

HiRDB/Single Server

Do not specify this option. A single server name will be ignored, if specified.

HiRDB/Parallel Server

Specify the name of the server used to create index information files.

(b) directory-name

~ <pathname> ((1-255))

Specifies the absolute path name of the directory in which index information files are to be created.

(3) Notes

The following shows the name of the index information file that is created automatically.

(a) When the inner replica facility is not used

directory-name/`INDEX`-*index-name*-*name-of-index-storage-RDAREA* -*unique-character-string*

Example:

If the `idxwork` statement specifies `/hd400` as the directory name, `IDX1` as the index name, and `USER01` as the name of the index storage RDAREA, the created index information file will have the following name:

```
/hd0400/INDEX-IDX1-USER01-aaaa00001
```

(b) When the inner replica facility is used

directory-name/`INDEX`-*index-name* -*name-of-index-storage-RDAREA*-`GN`*generation-number*-*unique-character-string*

Example:

If the `idxwork` statement specifies `/hd400` as the directory name, `IDX2` as the index name, `USER01` as the name of the original index storage RDAREA, and `2` as the generation number of the RDAREA to be processed, the created index information file will have the following name:

```
/hd0400/INDEX-IDX2-USER01-GN2-aaaa00002
```

8.9.8 sort statement (specification of work directory for sorting)

The `sort` statement specifies information about a work file for sorting that is used when an index is created.

Criterion

If possible, always specify the `sort` statement in order to avoid a shortage of space in the `/tmp` directory.

Rules

1. When the `sort` statement is omitted, the utility assumes the `/tmp` directory at the server that contains index storage RDAREAs.
2. You can specify as many `sort` statements as follows:

   HiRDB/Single Server

   Specify only one `sort` statement.

   HiRDB/Parallel Server

   For an index of a partitioned table, specify as many `sort` statements as there are servers that contain the table partitions. For a non-partitioned table or when reorganizing a partitioned table in units of RDAREAs, specify only one `sort` statement.

(1) Format

```
sort [server-name] directory-name[,buffer-size-for-sorting]
```

(2) Explanation

(a) server-name

~ <identifier> ((1-8))

Specifies the name of the server used to create the work file for sorting.

HiRDB/Single Server

Do not specify this option. A single server name will be ignored, if specified.

HiRDB/Parallel Server

Specify the name of the server used to create the work file for sorting.

(b) directory-name

~ <pathname> ((1-255))

Specifies the absolute path name of the directory in which sort work file is to be created.

(c) buffer-size-for-sorting

~ <unsigned integer> ((128-2097152)) <<1024>>

Specifies in KB the size of memory that is to be used as the buffer.

The system allocates this buffer at the single server for a HiRDB/Single Server and at the back-end server for a HiRDB/Parallel Server.

The sort process creates a temporary work file for sorting in a specified directory. You can use the following formula to determine a buffer size that minimizes the file size. This is just a guideline; avoid using a large value. Note that this formula provides a value that minimizes the file size, not a value that minimizes the sorting time. If there is enough memory, specify a buffer size of about several megabytes to several tens of megabytes.

• In 32-bit mode HiRDB

$$\text{Buffer size (bytes)} \geqq \frac{R+7}{2} + \sqrt{(B+8) \times n \times A + \frac{(R+7)^2}{4}} + C$$

• In 64-bit mode HiRDB

$$\text{Buffer size (bytes)} \geqq \frac{R+15}{2} + \sqrt{(B+8) \times n \times A + \frac{(R+15)^2}{4}} + C$$

n: Number of unloaded data items. For a repetition column, this is the number of elements, not the number of rows.

k: Key length (calculated as a maximum value). For details about how to determine the key length, see the example of calculating the number of index storage pages in the *HiRDB Version 8 Installation and Design Guide*.

x: 10, if all key component columns are fixed length; 12, if at least one of the key component columns is variable length.

c:

Number of columns composing the index

y:

For Linux version, 2; otherwise, 1.

z:

For a variable-length multicolumn index, $c \times 4$; otherwise, 0.

K:

For a variable-length multicolumn index, $k + c + 8$; otherwise, $k + 12$.

N:

For a variable-length multicolumn index, $(c \times 2) + y$; otherwise, $3 + y$.

R:

$k + x + z$

A:

For 32-bit mode HiRDB, $R + (K + 8) + 28$; for 64-bit mode HiRDB, $R + (K + 8) + 56$.

B:

For 32-bit mode HiRDB, $R + (K + 8) + 56$; for 64-bit mode HiRDB, $R + (K + 8) + 104$.

C:

For 32-bit mode HiRDB, $2092 + (N \times 32) + (K + 8)$; for 64-bit mode HiRDB, $2112 + (N \times 32) + (K + 8)$.

(3) Note

Do not allocate NFS to the directory that is specified in the `sort` statement. If NFS is allocated, twice as much space is required for the work file for sorting as when a local file is used. Also, problems occur related to retention of work files for sorting.

8.9.9 lobunld statement (specification of LOB data unload file)

The `lobunld` statement specifies information about a LOB data unload file to reorganize a table with LOB columns.

Criteria

Specify the `lobunld` statement to reorganize a table with LOB columns. However, do not specify this statement if either of the following is true:

- The `-j` option is specified.

- Only the LOB column structure base table is to be reorganized.

Rules

1. You can specify as many `lobunld` statements as follows:

   HiRDB/Single Server

   Specify only one `lobunld` statement regardless of the reorganization units (tables or RDAREAs).

   HiRDB/Parallel Server

   When reorganizing only the LOB columns of a row-partitioned table, specify as many `lobunld` statements as there are servers that contain the LOB column storage RDAREAs. When reorganizing a table in units of RDAREAs or organizing a non-partitioned table, specify only one `lobunld` statement.

   If the `-g` option is specified and table data is to be unloaded to a single LOB data unload file, specify only one `lobunld` statement.

2. For reorganization or reload operation with the synchronization point specification, you cannot specify the `lobunld` statement.

(1) Format

```
lobunld [{server-name:|host-name:}]
            LOB-data-unload-file-name
            [,LOB-data-unload-file-name]
            [EasyMT-information|HiRDB-file-information]
```

(2) Explanation

(a) server-name

~ <identifier> ((1-8))

Specifies the name of the server used to create the LOB data unload file.

HiRDB/Single Server

Do not specify this option for a HiRDB/Single Server.

HiRDB/Parallel Server

When reorganizing multiple LOB columns of a row-partitioned table, specify the name of the server used to create the LOB data unload file. When reorganizing a row-partitioned table in units of RDAREAs or reorganizing a non-partitioned table, there is no need to specify this option.

When unloading to a single LOB data unload file specifying the `-g` option, specify the name of the server used to create the LOB data unload file.

(b) host-name

～ <identifier> ((1-32))

Specifies the name of the host used to create the LOB data unload file.

HiRDB/Single Server

When creating the LOB data unload file on a utility special unit, specify the name of that host. When this option is omitted, the system creates the LOB data unload file on the unit where the single server is located.

HiRDB/Parallel Server

Do not specify this option for a HiRDB/Parallel Server.

(c) LOB-data-unload-filename

～ <pathname>

Specifies the absolute path name of the LOB data unload file.

You can use the following file as a LOB data unload file:

- Regular file

  You can create a regular file in a file system provided by the operating system. In this case, no preparations are necessary.

- Character special file

  Create a character special file in a HiRDB file system area for utilities that was created by the `pdfmkfs` command.

If you have specified the type of LOB data unload file in the `-f` option, specify the LOB data unload file names as follows.

When the LOB data unload file is EasyMT:

You can specify a maximum of two LOB data unload files.

If using MTguide, you can specify a device symbolic name or device group name managed by MTguide.

When the LOB data unload file is a HiRDB file:

- Express the name of the LOB data unload file as 1 to 167 characters.
- To use a HiRDB file, the HiRDB file system area must have already been created by the `pdfmkfs` command. In this case, specify `UTL` as the usage purpose in the `-k` option of the `pdfmkfs` command.
- An error results if the HiRDB file specified is in the HiRDB file system area that was created by the `pdfmkfs` command specifying a value other than

`UTL` in the `-k` option.

- If the HiRDB specified file name is not found in the HiRDB file system area, `pdrorg` creates a new HiRDB file. If you specify the name of an existing HiRDB file, the utility overwrites the file.

Note about LOB data unload files

1. If there is too much table data to fit in one LOB data unload file, you can specify multiple files.
2. When specifying multiple LOB data unload files, note the following:
   - You cannot use an NFS file.
   - You cannot use a direct tape unit without using EasyMT.

   The file attributes must be the same as follows:
   - EasyMT

     You need to initialize all volumes as EMTVOL_EL (EasyMT-labeled volumes).
   - HiRDB files

     You cannot mix regular files and character special files. Use either regular files only or character special files only.

(d) EasyMT-information

```
[file=filename]
[{,vol=volume-name|,vol=(volume-name[,volume-name]...)}]
[,bufno=buffer-sectors-count]
[,fileno={file-sequence-number|ADD}]
```

Specify this operand if the LOB data unload file is an EasyMT file.

By specifying a file name and volume name, you can check to see if the actual LOB data unload file and volume have the specified names. If you do not want to check the file or volume name, omit this operand.

If you are omitting a file name, do not specify a comma before the first specified item.

`file=`*filename* ~ <alphanumerics> ((1-17))

Specifies the name of the file.

`[{,vol=`*volume-name*`|,vol=(`*volume-name*`[,`*volume-name*`]...)}`

~ <alphanumerics> ((1-6))

Specifies the names of the volumes containing the file. You can specify up to 255 volume names in as much as they fit in one line.

`bufno=`*buffer-sectors-count* ~ <unsigned integer> ((1-256)) <<10>>

Specifies the number of buffer sectors to be used by EasyMT.

`fileno={`*file-sequence-number*`|ADD}`

Specifies the position of the LOB data unload file on the magnetic tape.

file-sequence-number ~ <unsigned integer> ((1-99)) <<1>>

If you have specified -k rorg or -k unld, the utility creates a file at the specified location and deletes any file following the specified location.

If you have specified -k reld, the utility reads the file at the specified location (if fileno is omitted, the utility reads the next file).

`ADD`

If you have specified -k rorg or -k unld, the utility creates the file at the end of the volume.

If you have specified -k reld, you cannot specify ADD.

(e) HiRDB-file-information

```
[init={initial-allocation-size
   |(initial-allocation-size[,initial-allocation-size]...)}]
[,incr=secondary-allocation-size
   |(secondary-allocation-size[,secondary-allocation-size]...)}
```

Specify this operand if the LOB data unload file is a HiRDB file. Be sure to specify the HiRDB file information on a single line.

If you omit `init`, do not specify a comma (`,`) before `incr`.

`init={`*initial-allocation-size*`|`
`(`*initial-allocation-size*`[,`*initial-allocation-size*`]...)}` ~ <unsigned integer> ((1-1048574)) <<100>>

Specify in MB the size of the area to be allocated to write to the HiRDB file.

Estimation formula:

For details about determining the initial allocation size, see the section in the *HiRDB Version 8 Installation and Design Guide* that describes the file size during execution of the database reorganization utility (`pdrorg`). Obtain the size of the LOB data unload file in bytes, then use the following formula:

Initial allocation size = ↑(size of LOB data unload file in bytes + 1024) ÷

(1024 × 1024)↑ (MB)

Rules

1. The initial allocation size must be smaller than the value specified in the `-n` option when the HiRDB file system area was created with the `pdfmkfs` command (because the system requires an area for management purposes).
2. Be sure to specify a nonzero value in the `-e` option of the `pdfmkfs` command. Otherwise, the secondary allocation is not possible.
3. Use the `pdfstatfs` command to determine the permitted of initial allocation size.
4. Specify as many allocation sizes as there are files specified in the `lobunld` statement. The system resolves a mismatch between the number of initial allocation sizes and the number of file names as follows:

Condition	Resolution
Number of files < number of initial allocation specifications	As many initial allocation specifications are valid as there are files, and subsequent initial allocation specifications are ignored.
Number of files = number of initial allocation specifications	Specification for each file is valid.
Number of files > number of initial allocation specifications	The last initial allocation specification is assigned to all the remaining files.

`incr=`*secondary-allocation-size*|
`(`*secondary-allocation-size*`[,`*secondary-allocation-size*`]...)}` ~ <unsigned integer> ((1-1048574)) <<10>>

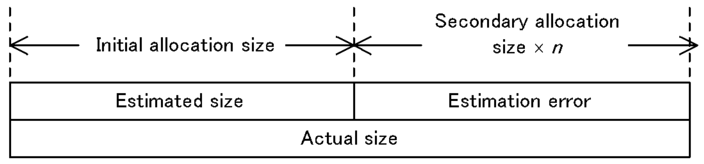
Specifies the size of the extension area when the initial allocation size is not sufficient during a write operation on the HiRDB file.

The utility does not expand the HiRDB file in the following cases:

- Attempt to expand the corresponding HiRDB file system area will exceed the specified number of extensions.
- Attempt to expand the corresponding HiRDB file will exceed the maximum number of extensions for individual HiRDB files (23 times).

If the actual size exceeds the estimated size, specify the secondary allocation size in such a manner that neither of the previous attempts is made. Figure 8-19 shows limitations to the secondary allocation size. Specify the value of *n* so that the above two attempts are avoided.

Figure 8-19: Limitations to the secondary allocation size (lobunld statement)



When creating a new HiRDB file system area, you can specify the number of secondary allocations in the `-e` option of the `pdfmkfs` command. For an existing HiRDB file system area, check the value of `available expand count` in the `pdfstatfs` command output result.

8.9.10 unlduoc statement (specification of UOC storage library information)

The `unlduoc` statement specifies information about a UOC that is to be used for database reorganization.

Criteria

Specify the `unlduoc` statement when a UOC is to be used for database reorganization.

Rules

You can specify only one `unlduoc` statement.

(1) Format

```
unlduoc entry=function-name
       [bloblimit=memory-space-allocation-size]
       [fixrow={y|n}]
```

(2) Explanation

(a) entry=function-name

Specifies the name of function to be called in the library.

(b) bloblimit=memory-space-allocation-size

~ <unsigned integer> ((1-2097152))

Specifies the maximum size in kilobytes of the memory space that is to hold the data to be passed to the UOC. If any of the following conditions is satisfied, the utility holds in memory all data to be passed to the UOC:

- When the table contains columns of an abstract data type and a constructor parameter reverse creation function corresponding to the `unld_func` statement is specified, the return value of the specified function has the large object data type.
- The table contains columns of the large object data type.

If there are multiple columns that satisfy these conditions, this operand applies to all such columns.

Criteria

Specify this option when any of the following conditions is satisfied:

- When the table contains columns of an abstract data type and a constructor parameter reverse creation function corresponding to the `unld_func` statement is specified, the return value of the specified function has the large object data type.

- A memory shortage may occur if the memory space is allocated for data based on the defined column lengths.
- The real data is clearly smaller than the defined column lengths.

Rules

1. For a column of the large object data type or if a defined column length is less than the memory space allocation size, the column length defined for that column becomes the memory space allocation size.
2. An error results if data exceeds the specified size (if the defined column length is less than the specified value or there is data that exceeds the defined column length).

(c) fixrow={y|n}

If a FIX table is to be processed, `pdrorg` specifies the storage method for the data to be passed to the UOC. Specifying `y` for a non-FIX table results in an error.

`y`:

Passes data in the order the columns are defined (consecutively). Specify `y` if you want to output all the data passed by `pdrorg` in the batch mode (you cannot directly reference any data type that requires alignment guarantee).

`n`:

Corrects the data start address according to the boundary of the data type and then passes the data. Specify `n` if you want to directly reference data (you can directly reference data without having to copy it to a separate area because the start address of the data type that requires alignment guarantee has already been corrected).

8.9.11 tblname statement (re-registration of table data to another table)

The `tblname` statement specifies that unloaded table data is to be reloaded to another table.

Criterion

Specify the `tblname` statement to migrate data between different tables on the same system or different systems.

Rules

1. You cannot specify the `tblname` statement in the following cases:

- A table to be reloaded or a table unloaded without the `-j` option specified has LOB columns or abstract data type columns including the LOB attribute.
- Table definitions do not match between the unloaded table and the table to be reloaded.

2. To specify the `tblname` statement, you must specify at least one `unload` statement. You cannot specify more than one `tblname` statement.

(1) Format

```
tblname {[authorization-identifier.]table-identifier|authorization-identifier}
```

(2) Explanation

(a) [authorization-identifier.]table-identifier

Specifies the authorization identifier and table identifier of the unloaded table.

For details about how to specify `[`*authorization-identifier*`.]`*table-identifier*, see the explanation of the `-t` option.

Rules

1. If the authorization identifier is omitted, the system utility assumes the authorization identifier specified in the `-t` option or the one used to establish connection with HiRDB. Therefore, if the unloaded table and the table to be reloaded have different authorization identifiers, be sure to specify this option.
2. To reload to a table with an abstract data type, some plug-ins require a constructor parameter reverse creation function during the unload operation.
3. If you specify `[`*authorization-identifier*`.]`*table-identifier*, you cannot specify `[`*authorization-identifier*`.] all` in the `-t` option.

(b) authorization-identifier

To reload table data to another table in units of schemas, this option specifies the authorization identifier.

If you are specifying this option, you must specify `[`*authorization-identifier*`.]all` in the `-t` option.

8.9.12 array statement (specification of row data output format for repetition columns)

To unload a table with repetition columns, the `array` statement specifies the row data output format for the repetition columns.

Criterion

When unloading a table with repetition columns, you can specify the `array` statement if the `-W` option is specified to output data to a DAT-format or extended DAT-format file.

Rules

1. This control statement is ignored if specified for a file that is not in DAT or extended DAT format.
2. If this control statement is omitted when a table with repetition columns is to be unloaded, the utility assumes `ff` as the row data output format.

(1) Format

```
array [elmtype=row-data-output-format]
```

(a) [elmtype=row-data-output-format]

～ <<`ff`>>

Specify either `ff` or `vv`.

`ff`

The utility outputs as many data items separated by separator characters as there are maximum number of elements specified for the corresponding column during table definition. For an element that has no data or a null value, the utility outputs only a separator character. If a column value is the null value, the utility outputs separator characters equal to the maximum number of elements.

To use the output data as the database load utility's (`pdload`'s) input file, specify an appropriate null value option to handle the null value.

For details about the data output format when `ff` is specified, see Section *8.3.3 Format of database load utility input files*.

`vv`

The utility outputs as many data items separated by separator characters as there are elements containing actual data, and the number of output elements at the top of the array data.

If a column value is the null value, the utility outputs 0 as the number of elements.

For details about the data output format when `vv` is specified, see Section *8.3.3 Format of database load utility input files*.

8.9.13 unld_func statement (specification of constructor parameter reverse creation function)

The `unld_func` statement specifies a constructor parameter reverse creation function that reverses the creation of data values for abstract data type during an unload operation.

Criterion

Specify the `unld_func` statement to reorganize a table with an abstract data type provided by a plug-in that has an unloading facility.

Rules

1. You can specify as many `unld_func` statements as there are abstract data types defined for table columns.
2. If you are specifying the `unld_func` statement along with `-k rorg`, also specify the `reld_func` statement.

(1) Format

```
unld_func type=[authorization-identifier.]abstract-data-type-name,
          func=function-name (argument-type[,argument-type...])
          [,func=function-name (argument-type[,argument-type...])...]
```

(2) Explanation

(a) type=[authorization-identifier.]abstract-data-type-name

Specifies the authorization identifier and name of the abstract data type.

Rules

1. If the authorization identifier is omitted, the authorization identifier of the user who defined the abstract data type (normally `MASTER`) is assumed.
2. If the authorization identifier or abstract data type name contains a lowercase letter or a space, enclose it in double quotation marks (").

(b) func=function-name (argument-type[,argument-type...])

Specifies the name and argument type of the constructor parameter reverse creation function. For details about the name and argument type of a constructor parameter reverse creation function, see the applicable plug-in manual.

function-name

Specify the name of the constructor parameter reverse creation function.

argument-type

Specify the data type of the argument of the constructor parameter reverse creation function. Specify the argument's data type in the following format:

Specification method	Remarks
`integer`	`INT32`
`smallint`	`INT16`

`char`
`nchar`
`mchar`
	`CHAR8`

`varchar`
`nvarchar`
`mvarchar`
	`p_pdb_varchar_t`

`float`	`DOUBLE64`
`smallflt`	`REAL64`
`blob`	`BLOB`
abstract-data-type-name	—

8.9.14 reld_func statement (specification of constructor function)

The `reld_func` statement specifies a constructor function that generates values of an abstract data type during a reload operation.

Criterion

Specify the `reld_func` statement to reorganize a table with an abstract data type provided by a plug-in that has an unloading facility.

Rules

1. You can specify as many `reld_func` statements as there are abstract data types defined for table columns.
2. If you are specifying the `reld_func` statement along with `-k rorg`, also specify the `unld_func` statement.

(1) Format

```
reld_func type=[authorization-identifier.]abstract-data-type-name,
       func=function-name (argument-type[,argument-type...])
```

(2) Explanation

(a) type=[authorization-identifier.]abstract-data-type-name

Specifies the authorization identifier and name of the abstract data type.

Rules

1. If the authorization identifier is omitted, the authorization identifier of the user who defined the abstract data type (normally `MASTER`) is assumed.
2. If the authorization identifier or abstract data type name contains a lowercase letter or a space, enclose it in double quotation marks (`"`).

(b) func=function-name (argument-type[,argument-type...])

Specifies the name and argument type of the constructor function. For details about the name and argument type of a constructor function, see the applicable plug-in manual.

function-name

Specify the name of the constructor function.

argument-type

Specify the data type of the argument of the constructor function. For details about the format of argument data type, see Section *8.9.13 unld_func statement (specification of constructor parameter reverse creation function)*. You cannot specify the name of an abstract data type.

8.9.15 constraint statement (specification of check pending status)

The `constraint` statement specifies settings for check pending status.

When a table for which check constraints or referential constraints have been defined is reorganized, the check pending status is not changed because constraint integrity can be guaranteed except when a row in the referenced table that is being referenced by a referencing table is deleted by a UOC. The check pending status is set only when a referenced table is reorganized by a UOC.

If reloading is performed, such as reloading using an old unload data file or reloading into a separate table, constraint integrity may no longer be guaranteed. Therefore, the applicable table and related referencing tables are placed in check pending status.

You can also specify a control statement that will prevent the check pending status from being changed (set). For details about check pending status, see the manual *HiRDB Version 8 Installation and Design Guide*.

Criteria

If integrity can be guaranteed, it is recommended that you specify the `constraint` statement (to not change (set) the table's check pending status). However, if any of the following conditions is applicable, it is recommended that you omit the `constraint` statement in order to maintain constraint integrity:

- Reloading using an old unload data file

  If data in a referencing table or referenced table is updated after it was unloaded, and reloading is performed using the unload data file that existed before the table was updated, data integrity may no longer be guaranteed between the referenced table and the referencing table because table data that existed before updating is being used.

- Reloading to another table or a table whose constraint definition has been changed

  When reloading is performed on a different table from the one used for unloading, data whose integrity cannot be guaranteed may be reloaded, because `pdrorg` does not perform data integrity checking.

- Reorganizing involving row deletion by a UOC

  When a UOC is used to reorganize a referenced table and to delete row data, integrity may be lost.

Rules

1. The `constraint` statement can be specified when reorganization (`-k rorg`) or reloading (`-k reld`) is performed, and when the value of the `pd_check_pending` operand in the system definition is `USE`.

If the value of the `pd_check_pending` operand in the system definition is `NOUSE` or if another facility (such as unloading (`-k unld`)) is used, this control statement is ignored if it is specified, in which case the table's check pending status is not changed.

(1) Format

```
constraint [pending=no]
         [ref_pending=no]
```

(2) Explanation

(a) pending=no

Specifies that when reorganization or reloading is to be performed on a referencing table or a table for which check constraints have been defined, the table's check pending status is not to be changed (set). For details about the default value that is assumed when this operand is omitted, see *(4) Range of check pending status settings*.

(b) ref_pending=no

Specifies that when reorganization or reloading is to be performed on a referenced table, the check pending status of a referencing table related to the referenced table is not to be changed (set). For details about the default value that is assumed when this operand is omitted, see *(4) Range of check pending status settings*.

(3) Notes

1. For notes about setting related referencing tables in check pending status when reorganization or reloading is performed on a referenced table, see *C. RDAREA Status During Command Execution*.
2. The current RDAREA is placed in check pending status when the inner replica facility is being used and the following are applicable: data reorganization or reloading is being performed on a referenced table, the referencing table related to the referenced table is to be placed in check pending status, and the replica RDAREA storing the referencing table is not in the generation specified in the `-q` option.
3. The table is placed in check pending status even when the reload count is 0 and constraint integrity can be guaranteed.
4. If reorganization involving the setting of check pending status results in an error and reorganization is re-executed, the table is placed in check pending status again.
5. When check pending status is set, the related resources are locked. Once the setting of the check pending status has been completed, the resources are

unlocked. For details about the lock, see *B.2 Lock mode for utilities*.

(4) Range of check pending status settings

Table 8-17 shows the `pdrorg` execution conditions and whether or not the check pending status is set.

Table 8-17: pdrorg execution conditions and whether or not the check pending status is set

Execution condition					**Whether or not check pending status is set**		
pd_check_pending operand value in the system definition	**Exe facility**	**Operand specification in the constraint statement**		**Use of UOC**	**Target table**		
					Ref'd table	**Ref'ing table**	**Table with check constraints defined**
USE	Reorg	`pending=no`	Yes	Yes	NC	NC	NC
				No	NC	NC	NC
			No	Yes	NC	NC	NC
				No	NC	NC	NC
		`ref_pending =no`	Yes	Yes	NC	NC	NC
				No	NC	NC	NC
			No	Yes	S	NC	NC
				No	NC	NC	NC
	Reload	`pending=no`	Yes	●	NC	NC	NC
			No	●	NC	S	S
		`ref_pending =no`	Yes	●	NC	NC	NC
			No	●	S	NC	NC
	Other	●	●	●	NC	NC	NC
Other	●	●	●	●	NC	NC	NC

Legend:

Exe facility: Executing facility

Ref'd table: Referenced table

Ref'ing table: Referencing table

Reorg: Reorganization

S: Check pending status is set.

NC: Check pending status is not changed (current status is maintained).

●: Not applicable

Table 8-18 shows the range of check pending status settings for check constraints during reorganization or reloading. Table 8-19 shows the range of check pending status settings for referential constraints.

Table 8-18: Range of check pending status settings for check constraints

Unit of execution	**Range of check pending status settings for check constraints**		
	Data dictionary table		**Table information in RDAREA**
	SQL_TABLES table CHECK_PEND2 column	**SQL_CHECKS table CHECK_PEND2 column**	**Check constraint status**
Table	C	C	T*
RDAREA	C	C	R*
Server	C	C	A*

Legend:

C: Sets the check pending status.

T: Sets table information (check constraint status) in all RDAREAs that store the table.

R: Sets table information (check constraint status) in the RDAREA that was specified in the `-r` option.

A: Sets table information (check constraint status) in all RDAREAs that store tables in the back-end servers in all `unload` statements specified.

* If the inner replica facility is used, the following generations are subject to check pending status setting:

- If the `-q` option is specified, the generation specified in the `-q` option is subject to setting.
- If the `-q` option is omitted, the current generation is subject to setting.

Table 8-19: Range of check pending status settings for referential constraints

Unit of execution	Target table					
	Referenced table			Referencing table		
	Check pending status of the referencing table related to referenced table			Check pending status of referencing table		
	Data dictionary table		TIRDA	Data dictionary table		TIRDA
	SQLTC	SQLTC	REFCS	SQLTC	SQLTC	REFCS
Table	C	C	T*	C	C	T*
RDAREA	C	C	T*	C	C	R*
Schema	C	C	T*	C	C	T*
Server	C	C	T*	C	C	A*

Legend:

TIRDA: Table information in RDAREA

SQLTC: `SQL_TABLES` table `CHECK_PEND` column

SQLRC: `SQL_REFERENTIAL_CONSTRAINS` table `CHECK_PEND` column

REFCS: Referential constraint status

C: Sets the check pending status.

T: Sets table information (check constraint status) in all RDAREAs that store the table.

R: Sets table information (check constraint status) in the RDAREA that was specified in the `-r` option.

A: Sets table information (check constraint status) in all RDAREAs that store tables in the back-end servers in all `unload` statements specified.

Note:

If a UOC is used, the check pending status is set.

* If the inner replica facility is used, the following generations are subject to check pending status setting:

- If the `-q` option is specified, the generation specified in the `-q` option is subject to setting. However, if reorganization or reloading is to be performed on a referenced table and the replica RDAREA storing the related referencing table is not in the generation specified in the `-q` option, the current RDAREA generation becomes subject to setting.

- If the `-q` option is omitted, the current generation is subject to setting.

(5) Example of whether or not check pending status can be set

This subsection describes whether or not check pending status can be set for tables `T1` through `T5` when the `pending` and `ref_pending` operands are specified in the `constraint` statement.

Description of the tables

`T1`: Table with the primary key (referenced table for `T2`)

`T2`: Table with the primary key and the foreign key that references `T1`'s primary key (referencing table for `T1` and referenced table for `T3`)

`T3`: Table with the foreign key that references `T2`'s primary key (referencing table for `T2`)

`T4`: Table for which check constraints have been defined

`T5`: Table with no constraint definition

Table 8-20 shows the table's check pending status when reloading is performed on tables `T1` through `T5`.

Table 8-20: Table's check pending status when reloading is performed

Table subject to reloading and whether or not there are constraint definitions				**Operand specification in the constraint statement**		**Check pending status for table**				
Table name	**Referential constraint**		**Check constraint**	**pend**	**ref_pend**	**T1**	**T2**	**T3**	**T4**	**T5**
	Referenced table	**Referencing table**								
T1	Y	N	N	Omit'd	Omit'd	●	P	●	●	●
					Spec'd	●	N	●	●	●
				Spec'd	Omit'd	●	P	●	●	●
					Spec'd	●	N	●	●	●
T2	Y	Y	N	Omit'd	Omit'd	●	P	P	●	●
					Spec'd	●	P	N	●	●
				Spec'd	Omit'd	●	N	P	●	●
					Spec'd	●	N	N	●	●

Table subject to reloading and whether or not there are constraint definitions				Operand specification in the constraint statement		Check pending status for table				
Table name	Referential constraint		Check constraint	pend	ref_pend	T1	T2	T3	T4	T5
	Referenced table	Referencing table								
T3	N	Y	N	Omit'd	Omit'd	●	●	P	●	●
					Spec'd	●	●	P	●	●
				Spec'd	Omit'd	●	●	N	●	●
					Spec'd	●	●	N	●	●
T4	N	N	Y	Omit'd	Omit'd	●	●	●	P	●
					Spec'd	●	●	●	P	●
				Spec'd	Omit'd	●	●	●	N	●
					Spec'd	●	●	●	N	●
T5	N	N	N	Omit'd	Omit'd	●	●	●	●	●
					Spec'd	●	●	●	●	●
				Spec'd	Omit'd	●	●	●	●	●
					Spec'd	●	●	●	●	●

Legend:

pend: pending

ref_ pend: ref_pending

Y: Defined

N: Not defined

Omit'd: Omitted

Spec'd: Specified

P: Check pending status is set.

N: Check pending status is not changed (the current status is maintained).

● : Not applicable (`pending` and `ref_pending` operands are ignored).

(6) Timing of setting check pending status

The target table subject to reorganization or reloading and the referencing tables related to that target table are placed in check pending status before reorganization or reloading is executed. If `pdrorg` rolls back due to an error after it has set the table's check pending status, the table is still placed in check pending status because setting of the check pending status has been completed (commit is completed).

If reorganization or reloading is performed for each schema, each target table is placed in check pending status before reorganization or reloading is executed. If the process that sets the check pending status on the table results in an error, `pdrorg` terminates with an error. Therefore, all tables processed before the occurrence of the error are placed in check pending status, but unprocessed tables are not placed in check pending status.

8.9.16 option statement (specification of data processing information)

The `option` statement specifies the optional data processing functions.

Criteria

Specify the `option` statement to use any of the following optional functions:

- Specifying the space conversion level (`spacelvl` operand)
- Changing the percentage of free space during a reload operation (`tblfree` operand)
- Changing the percentage of free space during index creation (`idxfree` operand)
- Data reorganization or reloading with the synchronization point specification (`job` operand)
- Monitoring the `pdrorg` execution time (`exectime` operand)
- Forcibly placing the table in normal status during re-execution of table reorganization (`tblstatus` operand)
- Changing the lock mode of the table to be unloaded (`unldenq` operand)

(1) Format

```
option spacelvl = {0|1|3}
      [tblfree = {percentage-of-unused-space|
      ([percentage-of-unused-space],
      percentage-of-free-pages-in-segment)}]
      [idxfree = percentage-of-unused-area]
      [job = [job-name], [lines-count-between-synchronization-points]
      [, CLR]]
      [exectime=pdrorg-execution-monitoring-time]
      [tblstatus=clear]
      (unldenq={tblenq|rdenq|nowait})
```

(2) Explanation

(a) spacelvl = {0|1|3}

Specifies whether or not to execute space conversion on data during reorganization. Available space conversion levels are 0, 1, and 3.

`0`

The system does not execute space conversion on the data.

`1`

The system executes space conversion on the unload data during a reload operation.

Criteria

Specify this space conversion level in the following cases:

- To use the same space code consistently throughout the existing data
- When migrating table data to another system, the same space code is to be used consistently in the target system.

`3`

The system executes the same space conversion as for `1` and the space conversion on unload data during an unload operation specifying the `-W` option.

Criterion

Specify this space conversion level to use the same space code consistently through the unload data, such as when using a UAP to process the unload data that is output with the `-W` option specified.

Notes

- The following table shows whether or not space conversion is executed depending on the specification:

Type of pdrorg processing	**Option**			**Value of spacelvl**		
	-c	**-k**	**-W**	**0**	**1**	**3**
Reorganization	`user`	`rorg`	—	N	Y[1]	Y[1]
Unloading	`user`	`unld`	Specified	N	N	Y[2]
			Not specified	N	N	N
Reloading	`user`	`reld`	—	N	Y[1]	Y[1]
Data dictionary table reorganization	`dic`	`rorg`	—	N	N	N

Y: Space conversion is executed.

N: Space conversion is not executed.

— : Not applicable.

[1] If a table column being processed has the national or mixed character string type, the system converts space in the unload data during a reload operation as follows:

National character string type

The system converts two consecutive single-byte spaces in the unload data to one double-byte space, in units of two bytes from the top.

Mixed character string type

The system converts each double-byte space in the unload data to two consecutive single-byte spaces.

[2] If a table column being processed has the national character string type, the system converts each double-byte space in the unload data to two single-byte spaces during an unload operation.

When the character codes are `utf-8`, the system converts one double-byte space (3 bytes) to two single-byte spaces. For `MCHAR`, the system sets trailing single-byte spaces up to the definition length. For `MVARCHAR`, the data length remains shortened.

- If the table contains an abstract data type, space conversion depends on the abstract data type reloading method shown as follows:

Abstract data type reloading method	**Constructor function's argument data type**	**Value of spacelvl in option statement**	
		Omitted or 0	**1 or 3**
Reloading from the unload data file obtained by the plug-in's unload facility	—	N	N

Abstract data type reloading method	**Constructor function's argument data type**	**Value of spacelvl in option statement**	
		Omitted or 0	**1 or 3**
Using a constructor function to reload from the unload data file obtained by the plug-in's constructor parameter reverse creation function	`NCHAR`, `NVARCHAR`, `MCHAR`, `MVARCHAR`	N	Y*
	Other	N	N
Reloading without using any of these facilities	—	N	N

Y: Space conversion is executed.

N: Space conversion is not executed.

— : Not applicable.

* Space conversion is executed for the data type of the constructor function argument (argument type specified in the `func` operand of the `reld_func` statement for the corresponding abstract data type).

- If the column used as the cluster key or index key contains a national character string type and the data is to be unloaded in the order of key values, the system unloads the data in the order of their values stored in the database, regardless of the `spacelvl` option specification. Therefore, if space conversion is executed during an unload operation specifying the `-W` option, the output data may not be sorted in the order of key values. In this case, you can sort data in the order of key values by reorganizing the data without space conversion after the data load operation.

  In the case of a table partitioned and stored at multiple servers, the data is not always unloaded in the order of key values regardless of the execution of space conversion, as is the case with the data unload operation in the order of cluster key or index key values. To store data in the order of key values, you need to sort the data after the unload operation.

- If you specify `spacelvl=1` or `spacelvl=3` in the `option` statement to reorganize a database that contains more than one kind of space code, the following events occur:
  - The data unloaded in the order of key values may no longer be sorted in the order of key values. In this case, the data can be stored in the order of key values by reorganizing it again without space conversion.
  - If the column used as the unique key index or primary key index is subject to space conversion, duplicate key values may result in the index. In this case, determine whether or not it must be a unique key or primary key and modify the corresponding data (such as by changing the key value or deleting

unneeded rows) before executing space conversion.

- If the column used as the partitioning key is subject to space conversion, the storage RDAREA may change, depending on the partitioning conditions. In this case, specify the `-g` option so that data can be processed even if its storage location changes. If the table contains LOB columns, specify the `-g` and `-j` options.

(b) tblfree = {percentage-of-unused-space| ([percentage-of-unused-space], percentage-of-free-pages-in-segment)}

Specify this operand to change the percentage of free space specified with `CREATE TABLE` (value of `PCTFREE`) for storing data during a reload operation.

For the reorganization in units of schemas, the specified value takes effect on all tables.

Criteria

Suppose that you have specified a non-zero value as the percentage of free space during table definition, and a space shortage has occurred while processing this table.

In this case, if you execute `pdrorg` on the table in the RDAREA with no free space, a shortage of RDAREA may occur because the defined percentage of free space takes effect during the execution of `pdrorg`. In this case, if you specify the `tblfree` operand, you can complete the `pdrorg` processing without temporarily expanding the RDAREA.

percentage-of-unused-space

You can specify a value in the range from 0 to 99.

percentage-of-free-pages-in-segment

You can specify a value in the range of 0 to 50.

(c) idxfree = percentage-of-unused-area

If you are creating indexes, specify this operand to change the percentage of unused area specified with `CREATE INDEX` (value of `PCTFREE`). The permitted value range is from 0 to 99.

For the reorganization in units of schemas, the specified value takes effect on all tables.

Criteria

Suppose that you have specified a non-zero value as the percentage of unused space during index definition and a space shortage has occurred while processing this table. In this case, if you execute `pdrorg` on the indexes in the RDAREA with no free space, a shortage of RDAREA may occur because the defined percentage of free area takes effect during the execution of `pdrorg`. In this case, if you specify the `idxfree` operand, you can complete the processing without temporarily expanding the RDAREA.

(d) job = [job-name], [lines-count-between-synchronization-points][, CLR]

Specify this operand to execute reorganization or reloading with the synchronization point specification.

Reorganization with the synchronization point specification or reload operation with the synchronization point specification is a method of reorganizing or reloading data in which a transaction is settled each time the specified number of data items are stored. If an error occurs while data is stored, this method enables you to restore the database in a short period of time without having to store data again from the beginning.

job-name ~ <alphanumerics> ((1-3))

Specify the name of the job that executes reorganization or reload operation with the synchronization point specification.

This job name is used during re-execution in the event the utility terminates abnormally during reorganization or during a reload operation. If this job name is the same as a `pdrorg` job name, a malfunction results. Therefore, make sure that the specified job name is unique in the HiRDB system.

lines-count-between-synchronization-points ~ ((1-1000)) <<100>>

Specify the number of data items as an interval at which a transaction is to be settled. For example, a value of 100 sets a synchronization point at every one million data items.

A small value settles a transaction after a small number of data items; therefore, rollback processing would take a short time in the event of abnormal termination. However, transaction generation and settlement and process restart occur frequently, resulting in a large overhead. On the other hand, a large value has a little effect on the performance, but the error recovery time would be long.

An error results if you specify only the number of lines between synchronization points without specifying a job name.

`CLR`

Specify this operand if there is no need to re-execute `pdrorg` or if `pdrorg` is to be re-executed after clearing the current synchronization point information in the data base.

If the utility terminates abnormally, the synchronization point information remains in the database. If this information is retained during the next execution, the utility may not function correctly.

Note that if you specify `CLR`, the system clears all synchronization point information, whether it was specified by another user or utility.

To determine if the synchronization point information is present in the database, use the database condition analysis utility (`pddbst`) to execute condition analysis in units of tables.

Criterion

This option is useful for loading a large amount of data, which takes a considerable amount of time to store the data. However, this has adverse effects on performance due to synchronization point processing. Additionally, for data pages, the number of pages required for storing data is greater than with normal reorganization because data storage begins in a new page at each synchronization point. Therefore, if there is not much free space in the RDAREA, we recommend that you do not use this option.

Example of job operand specification

- Reorganizing or reloading data with a synchronization point specified for every one million data items

  `job=JOB,100` or `job=JOB`

- Clearing the current synchronization point information and specifying new synchronization points at every one million data items to execute reorganization or reload operation

  `job=JOB,100,CLR` or `job=JOB,,CLR`

- Clearing the current synchronization point information and executing normal reorganization or reload operation

  `job=,,CLR`

Notes

1. Reorganization with the synchronization point specification is not applicable to data dictionary tables.
2. Synchronization point information is managed by the job name. Suppose that `pdrorg(A)` was executing reorganization or a reload operation with the synchronization point specification and it terminates abnormally. If you execute `pdload(B)` specifying the same job name for the same table, `pdload(B)` uses the synchronization point information that was left by `pdrorg(A)`. Therefore, the job names must be controlled so that they are not duplicated.
3. If reorganization or reload operation with the synchronization point specification terminates abnormally, you cannot execute the normal reorganization or reload operation (without the synchronization point specification) on the same table until you re-execute the reorganization or reload operation with the synchronization point specification with the same job name.
4. The synchronization point information is managed in units of RDAREAs. For a

row partitioned table, inconsistency occurs on the synchronization point information for the entire table in the following cases:

- Only some of the RDAREAs were re-initialized.
- An error occurred during reorganization or reload operation with the synchronization point specification in units of RDAREAs.

If the job name is the same, but the number of lines does not match, you cannot use the synchronization point information managed by each RDAREA for re-execution. If some of the RDAREAs have synchronization point information while the others do not, you can re-execute data loading using the retained synchronization point information. The following shows how the system handles reorganization or reload operation when the job name for synchronization point information is the same but the number of lines does not match:

Retained synchronization point information (number of lines between synchronization points)			**Input begin line for reorganization or reload operation with synchronization point specification in units of tables**
RDAREA 1	**RDAREA 2**	**RDAREA 3**	
0	0	0	Item 0
0	*n*	0	Item *n*
n	0	*m*	Cannot be executed.
n	*m*	*p*	Cannot be executed.

0: No synchronization point information retained

n, *m*, *p*: Retained number of lines between synchronization points:

1. Reorganization or reload operation with the synchronization point specification is not recommended if all the following conditions are met:
   - HiRDB/Parallel Server is used.
   - The table is partitioned and stored at multiple servers.
   - The `-g` option is omitted.

   In this case, processing takes place at each server; therefore, in the event of an error, some servers may be unloading while others are reloading or some server processing may have already been terminated. In such cases, the user must specify an appropriate re-execution method for each server in the event of abnormal termination, resulting in complex operations.
2. If reorganization or reload operation with the synchronization point specification terminates abnormally, you cannot modify the index creation method during re-execution (`-i` option).

3. If EasyMT is remounted before a reorganization or reload operation with synchronization point specification terminates abnormally, mount volume one again during the re-execution.
4. If reorganization or reload operation with the synchronization point specification terminates abnormally, and you re-initialize RDAREAs or execute the `PURGE TABLE` statement before re-execution, the existing synchronization point information is deleted.
5. If you are executing reorganization or reload operation with synchronization point specification only on a LOB column structure base table and the processing terminates abnormally, specify `CLR` during the re-execution. Otherwise, the search performance on the reorganized LOB columns may be reduced.
6. The maximum number of data items that can be controlled during reorganization or reloading with the synchronization point specification is 4,294,960,000. To reorganize more data than this amount, consider one of the following operations:
   - Reorganizing in units of RDAREAs
   - Creating multiple input data files for `pdload` by specifying `-W` and then performing data loading with synchronization point specification more than once with `pdload`

(e) exectime=pdrorg-execution-monitoring-time

~ <unsigned integer> ((1-35791394)) <<0>>

Specifies the `pdrorg` execution monitoring time in minutes. If you omit this operand, the utility does not monitor the execution time. If `pdrorg` processing does not terminate within the specified monitoring time, the utility terminates the `pdrorg` process forcibly and collects error information to investigate the cause of the no-response.

This operand takes precedence over the `pd_utl_exec_time` operand in the system definition.

Criterion

If you specify a monitoring time for nighttime batch processing, and `pdrorg` is placed in no-response status due to an error, such as a communication error (including a transient error) or a disk error, `pdrorg` will terminate abnormally. This enables you to detect an error and start recovery processing at an early stage.

Guideline for the specification value

The purpose of this operand is to detect a no-response error, not to monitor the execution time of a long transaction. Therefore, the specified time must be sufficient for processing the applicable table. For example, to monitor the execution time of a `pdrorg` that should terminate in 7-8 minutes, specify `exectime=20`, not `exectime=10`. If the amount of data increases proportionally, the specified value should be re-evaluated as needed.

(f) tblstatus=clear

Specify this option to forcibly place the table in the normal status during re-execution of table reorganization. In such a case, table reorganization will not restart.

Criteria

Specify this option to forcibly place the table in normal status such, as when you want to use SQL statements to reference an RDAREA (table) that has been placed in reload-not-completed data status.

Notes

1. You can specify this operand only when you specify `-k rorg`. If you specify this operand at any other time, an error results.
2. If you specify this operand to place the table in normal status and then you reorganize the table, reorganization will not restart. In such a case, you must restore the table from its backup copy. If you reorganize the table without restoring it first, the unload data file is overwritten and the number of data items becomes invalid (zero).

(g) unldenq={tblenq|rdenq|nowait}

This option specifies whether or not other transactions are to be permitted to update the table during unloading (changes the lock mode of the table that is to be unloaded). Note that only the areas of the table that are not being accessed can be updated during unloading.

`tblenq`

Allows other transactions only to reference the table. In this case, the table is placed in the protected retrieval mode and the RDAREA is placed in the shared retrieval mode.

`rdenq`

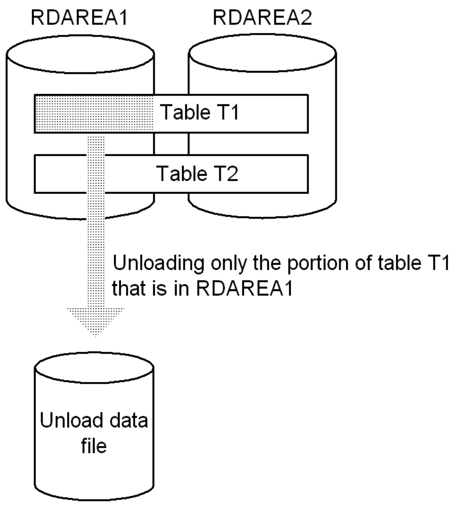
Allows other transactions only to reference the RDAREA. In this case, the table is placed in the shared retrieval mode and the RDAREA is placed in the protected retrieval mode.

`nowait`

Allows other transactions to reference the table and update the RDAREAs that are not subject to unloading. In this case, only the RDAREA is placed in the shared retrieval mode, enabling the table being unloaded to be updated. Use this operand only when there is a guarantee that the table to be unloaded (RDAREA) will not be updated, because if data loading or reloading is performed using the unload data file created by unload processing at the same time that update processing is being performed, data duplication or data loss may occur.

Example

If only table `T1` in `RDAREA1` (shaded part) is unloaded with the table and RDAREA layout shown below, whether or not other transactions can update the table and RDAREA depends on the operand specification.



Explanation

The following table shows whether or not other transactions can update the table and RDAREA:

unldenq value	Lock mode			Whether or not other transactions can execute							
	RDAREA	Table	NW TBL	Table T1				T2			
				RDAREA1		RDAREA2		RDAREA1		RDAREA2	
				REF	UP	REF	UP	REF	UP	REF	UP
`tblenq`	SR	PR	●	Y	N	Y	N	Y	Y	Y	Y
`rdenq`	PR	SR	●	Y	N	Y	Y	Y	N	Y	Y
`nowait`	SR	●	SR	Y	N*	Y	Y	Y	Y	Y	Y

Legend:

NW TBL: NOWAIT Table

REF: Referencing

UP: Updating

SR: Locked in the shared retrieval mode

PR: Locked in the protected retrieval mode

•: Not locked

Y: Executable

N: Not executable

* The user must ensure that data for table `T1` in `RDAREA1` will not be updated.

8.9.17 report statement (specification of a file to which pdrorg tuning information is output)

The `report` statement specifies a file to which the tuning information is to be output during the execution of `pdrorg`.

Criterion

Specify the `report` statement if you want to output the tuning information.

Rules

1. You can specify only one `report` statement.
2. The system does not output the tuning information unless the `report` statement is specified.

(1) Format

```
report file=process-results-filename
```

(2) Explanation

(a) process-results-filename

~ <pathname>

Specifies the absolute path name of the file to which the tuning information is to be output.

This file must be located at the host where `pdrorg` is executed.

(3) Output format

(a) Tuning information

```
pdrorg VV-RR(Object Option) *** DB REORGANIZATION *** 2003-03-31 11:06:07 ...........1

*** statistics report ***

execute time :   0: 1:46 ............................................................2
prepare time :   0: 0: 0 ............................................................3


*** shell/server information ***

name       cpu svrup(first) reciv  wait  sort  send unlod ixunl delet dtlod ixlod unldr
unldw(expnd) unput dtins ............................................................4
● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ● ● ● ● ● ● ● -
● ● ● ● ● ● ● ● ● ● ● ●
pdrorg       0     0(    0) ***** ***** ***** ***** ***** ***** ***** ***** ***** *****
*****(*****) ***** *****
pdrorgm      8     7(    7)    27 ***** ***** 12.5k ***** ***** ***** ***** *****  248k
 124k(*****)     3 *****

bes1        24 *****(*****)    14     0     2 6.24k     6 *****     1    16    57 *****
*****(*****) *****    16
bes2        26 *****(*****)    10     0     2 6.24k     6 *****     1    18    57 *****
*****(*****) *****    19
● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ● ● ● ● ● ● ● -
● ● ● ● ● ● ● ● ● ● ● ●
total       58     7(    7)    51     0     4 25.0k    12 *****     2    34   114  248k
 124k(*****)     3    35 ............................................................5


*** buffer information ***

server   maxio minio sumio brreq bwreq bfhit hitrt  read write lrreq lbrht lbrrt lwreq
lbwht
lbwrt lread lwrit flush bfupd bfred bfwrt cinsm cafls cafwr cfmax cfavg ldirc ldiuc
ldihc
ldird ldiwt lbfshc ..................................................................6
● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ● ● ● ● ● ● ● ● -
● ● ● ● ● ● ● ● ●
bes1        0     0    53 79.9k 44.3k 60.7k 76.1% 6.95k 12.2k     0     0 *****     0     0
*****     0     0 16.5k     0     0     0     0     0     0     0 1.00k 11.4k 11.4k     0
    0 11.4k 10.6k
bes2        0     0    52 79.9k 44.3k 60.8k 76.1% 6.95k 12.2k     0     0 *****     0     0
*****     0     0 16.5k     0     0     0     0     0     0     0 1.00k 11.4k 11.4k     0
    0 11.4k 10.6k
```

```
● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ● ● ● ● ● ● ● ●-
● ● ● ● ● ● ● ● ●
total       0     0   105  160k 88.6k  121k 76.1% 13.9k 24.3k     0     0 *****     0     0
*****     0     0 33.1k     0     0     0     0     0     0     0 1.00k 22.8k 22.8k     0
    0 22.8k 21.3k

pdrorg terminated, return code=0 ....................................................7
```

Explanation

1. Header information (version number and `pdrorg` start time)
2. `pdrorg` processing time
3. `pdrorg` preprocessing time
4. Each server information:

   `name`: Shell name or server name

   `cpu`: CPU processing time (seconds)

   `svrup(first)`: Server startup time (initial server access time) (seconds)

   `reciv`: Total wait time for communication from other processes (seconds)

   `wait`: Total time required for allocating locked resources (seconds)

   `sort`: Sorting time (seconds)

   `send`: Number of communications to other processes

   `unlod`: Unloading time (seconds)

   `ixunl`: Time spent to create index information files (applicable to index re-creation or index reorganization) (seconds)

   `delet`: Data deletion time (seconds)

   `dtlod`: Data loading time (seconds)

   `ixlod`: Time spent to create indexes from index information files (seconds)

   `unldr`: Unload data file input count

   `unldw(expnd)`: Unload data file output count

   `unput`: Communication message processing time during unload operation (seconds)

   `dtins`: Communication message processing time during reload operation (seconds)

5. Total value for each server information
6. Each buffer information

   `server`: Name of back-end server

   `maxio`: Maximum input/output time (seconds)

   `minio`: Minimum input/output time (seconds)

   `sumio`: Total input/output time (seconds)

   `brreq`: Number of `READ` requests for data and index buffer

   `bwreq`: Number of `WRITE` requests for data and index buffer

   `bfhit`: Data and index buffer hit count

   `hitrt`: Data and index buffer hit ratio (%)

   `read`: Data and index buffer `READ` count

   `write`: Data and index buffer `WRITE` count

   `lrreq`: Number of `READ` requests for LOB buffer

   `lbrht`: LOB buffer `READ` hit count

   `lbrrt`: LOB buffer `READ` hit ratio (%)

   `lwreq`: Number of `WRITE` requests for LOB buffer

   `lbwht`: LOB buffer `WRITE` hit count

   `lbwrt`: LOB buffer `WRITE` hit ratio (%)

   `lread`: LOB buffer `READ` count

   `lwrit`: LOB buffer `WRITE` count

   `flush`: Buffer flush count

   `bfupd`: `READ` wait count during buffer updating

   `bfred`: `READ` wait count during buffer `READ`

   `bfwrt`: `WRITE` wait count during buffer `WRITE`

   `cinsm`: Internal information used by the system

   `cafls`: Internal information used by the system

   `cafwr`: Internal information used by the system

   `cfmax`: Internal information used by the system

   `cfavg`: Internal information used by the system

   `ldirc`: Number of times the random access local buffer was used to reference index pages

ldiuc: Number of times the random access local buffer was used to update index pages

ldihc: Of bfhit, buffer hit count for index pages in the random access local buffer

ldird: Of read, the number of real READs for index pages in the random access local buffer

ldiwt: Of write, the number of real WRITEs for index pages in the random access local buffer

ldfshc: Of flush, number of times the random access local buffer was flushed

7. pdrorg's return code

Notes

1. Each item is displayed with a maximum length of three digits. If a value exceeds three digits, the system rounds up the fourth digit and displays the value with a decimal point and one of the units shown as follows:

   k: Kilo ($\times 10^3$ for time and count)

   M: Mega ($\times 10^6$ for time and count)

   G: Giga ($\times 10^9$ for time and count)

2. If there is no corresponding output information, the system displays *****.
3. The processing time less than one second is rounded off. This means that a value less than one second is displayed as 0 second.

(b) Processing result of reorganization in units of schemas

-k unld

```
pdrorg VV-RR(Object Option) *** DB REORGANIZATION *** 2000-11-16 13:27:36 .......1

*** schema unload processing list ***

schema name : USER01 ............................................................2
```

```
No.1     table name : TABLE11                    generation : 01 ............3

         rdarea name ...........4           row count ...........................5
         RD011                                       31,267
         RD021                                       31,540
         RD012                                       31,167
         RD013                                       31,169
         RD014                                       31,363
         RD022                                       31,366
         RD023                                       31,024
         RD024                                       31,104
                                  total       250,000 ...........................6

         return code=0 .........................................................7


No.2     table name : TABLE21                     generation : 01

         rdarea name                              row count
         RD011                                        1,246
         RD021                                        1,310
         RD012                                        1,264
         RD013                                        1,300
         RD014                                        1,286
         RD022                                        1,228
         RD023                                        1,165
         RD024                                        1,201
                                  total        10,000

         return code=0


 [Tuning information of (a)]

pdrorg terminated, return code=0 ...............................................8
```

Explanation

1. Header information (version number and `pdrorg` start time)
2. Schema name
3. Table identifier and generation number (the generation number is displayed only when the inner replica facility is used)
4. Table storage RDAREA name
5. Number of rows unloaded per RDAREA
6. Number of rows unloaded per table
7. Return code when one table was unloaded
8. `pdrorg`'s return code

-k reld

```
pdrorg VV-RR(Object Option) *** DB REORGANIZATION *** 2000-11-16 17:02:01 .......1

*** schema reload processing list ***

schema name : USER01 ..........................................................2


No.1     table name : TABLE11                    generation : 01 ...........3

         rdarea name ...........4            row count ..........................5
         RD011                                    31,267
         RD021                                    31,540
         RD012                                    31,167
         RD013                                    31,169
         RD014                                    31,363
         RD022                                    31,366
         RD023                                    31,024
         RD024                                    31,104
                                 total       250,000 ..........................6


*** index information file assign list *** ....................................7

index INDEX11 RD011 /tmp/INDEX-TABLE11-RD011-aaaa14376
index INDEX11 RD021 /tmp/INDEX-TABLE11-RD021-aaaa14380
index INDEX11 RD012 /tmp/INDEX-TABLE11-RD012-baaa14376
index INDEX11 RD013 /tmp/INDEX-TABLE11-RD013-caaa14376
index INDEX11 RD014 /tmp/INDEX-TABLE11-RD014-daaa14376
index INDEX11 RD022 /tmp/INDEX-TABLE11-RD022-baaa14380
index INDEX11 RD023 /tmp/INDEX-TABLE11-RD023-caaa14380
index INDEX11 RD024 /tmp/INDEX-TABLE11-RD024-daaa14380

         return code=0 .........................................................8


No.2     table name : TABLE12                        skipped.....................9

         return code=4


No.3     table name : TABLE21                        generation : 01

         rdarea name                            row count
         RD011                                     1,246
         RD021                                     1,310
         RD012                                     1,264
         RD013                                     1,300
         RD014                                     1,286
         RD022                                     1,228
         RD023                                     1,165
         RD024                                     1,201
                                 total        10,000
```

```
*** index information file assign list ***

index INDEX21 RD011 /tmp/INDEX-TABLE21-RD011-aaaa14378
index INDEX21 RD021 /tmp/INDEX-TABLE21-RD021-aaaa14745
index INDEX21 RD012 /tmp/INDEX-TABLE21-RD012-baaa14378
index INDEX21 RD013 /tmp/INDEX-TABLE21-RD013-caaa14378
index INDEX21 RD014 /tmp/INDEX-TABLE21-RD014-daaa14378
index INDEX21 RD022 /tmp/INDEX-TABLE21-RD022-baaa14745
index INDEX21 RD023 /tmp/INDEX-TABLE21-RD023-caaa14745
index INDEX21 RD024 /tmp/INDEX-TABLE21-RD024-daaa14745

         return code=0


 [Tuning information of (a)]

pdrorg terminated, return code=4 ...............................................10
```

Explanation

1. Header information (version number and `pdrorg` start time)
2. Schema name
3. Table identifier and generation number (the generation number is displayed only when the inner replica facility is used)
4. Table storage RDAREA name
5. Number of rows reloaded per RDAREA
6. Number of rows reloaded per table
7. Information about index information files (applicable if `-i n` is specified)

   `index` *index-identifier index-storage-RDAREA-name index-information-filename*
8. Return code when one table was reloaded
9. Indicates that reload processing was skipped, in which the system outputs the return code when one table was reloaded
10. `pdrorg`'s return code

-k rorg

```
pdrorg VV-RR(Object Option) *** DB REORGANIZATION *** 2000-11-16 13:57:08 .......1
  [Information for -k unld]
   [Information for -k reld]*
  [Tuning information of (a)]
pdrorg terminated, return code=0 ................................................2
```

* The schema name is not output.

Explanation

1. Header information (version number and `pdrorg` start time)
2. `pdrorg`'s return code

8.9.18 blobtovarchar statement (specification of column subject to data conversion)

The `blobtovarchar` statement specifies the name of an abstract data type column with `BLOB` attribute to be converted to the column `VARCHAR` type before unloading it.

Criterion

Specify the `blobtovarchar` statement to convert a column of abstract data type with `BLOB` attribute to the `VARCHAR` type before unloading it. This statement is applicable only to the abstract data type for which a constructor parameter reverse creation function returns a value of `BLOB` type.

Rules

1. You can specify only one `blobtovarchar` statement.
2. The `blobtovarchar` statement is applicable only to an unload operation specifying `-W bin`.
3. A data conversion error occurs if the size of data is 32 KB or greater (the maximum size of data definable as `VARCHAR` type is 32 KB).

(1) Format

```
blobtovarchar column-name[,column-name]...
```

(2) Explanation

(a) column-name

Specifies the name of an abstract data type column that has a constructor parameter reverse creation function.

The system treats a column name enclosed in double quotation marks as case sensitive; otherwise, the system treats it as all uppercase letters.

You can specify a column name for which the constructor parameter reverse creation function returns a value of `BLOB` type. Specifying any other column results in an error.

8.9.19 fixtext_option statement (specification of output data during creation of input data file in fixed-size data format)

When an input data file in the fixed-size data format is created with the `-W fixtext` option, the `fixtext_option` statement specifies information about the output data.

Criteria

When an input data file in the fixed-size data format is created (with the `-W fixtext` operand), the `fixtext_option` statement is specified in order to edit the output data.

Rules

1. Only one `fixtext_option` statement can be specified.
2. When you specify multiple operands, delimit them with the space character (`0x20`).
3. Do not specify the `fixtext_option` statement without any operand specified.

(1) Format

```
fixtext_option [cntlcode={nocheck|check|replace[,replacement-character]}]
               [enclose=enclosing-character]
               [format=data-type,output-format]
```

(2) Explanation

(a) cntlcode={nocheck|check|replace[,replacement-character]}

For the character string data, national character data, and mixed character string data types, specifies how to control data output when the character string data contains control characters (ASCII codes `0x00` through `0x1f`).

`nocheck`:

Output the column data in the database as is without checking for control characters. You specify this operand in order to output the column data in the database as is.

`check`:

Check for control characters; if the column data contains a control character, do not output the corresponding row to the unload data file. You specify this operand to suppress output of rows that contain control characters.

`replace[,`*replacement-character*`]`:

Check for control characters; if the column data contains a control character, replace it with the specified replacement character and then output the corresponding row to the unload data file. You specify this operand to output column data containing control characters in such a manner that the control characters can be viewed by a program such as Text Viewer.

Specify the replacement character as a 1-byte character.

Specification rules for the replacement character

1. When you omit the replacement character, the system assumes a colon.
2. None of the following characters can be specified as the replacement character:

   space, tab, asterisk, underscore
3. None of the following characters is suitable for the replacement character because they may be the same as codes used in the output data:
   - Characters used in the output data
   - Numeric data sign characters (+ and -)
   - Characters whose code is the same as that of a multi-byte character (|, ¥, [,], <, >, {, }, ￣)
   - Hyphen used in date interval, time interval, and time stamp data
   - Colon used in time and time stamp data
   - Period used in numeric value, date interval, time interval, and time stamp data

Rules for the cntlcode operand

1. When the `cntlcode` operand or `fixtext_option` statement is omitted, `cntlcode=nocheck` is assumed.
2. You can specify only one `cntlcode` operand in the `fixtext_option` statement.
3. The following table explains the relationships between whether or not there are control statements in column data, the `cntlcode` operand value, and the `pdrorg` processing:

Condition		pdrorg's processing	
Whether or not there are control characters in column data	cntlcode operand value	Checking for control characters	Data output method
Yes	`check`	Y*	Resumes processing without outputting the corresponding row (`pdrorg` terminates normally with return code `4`).
	`replace`		Replaces the control characters and then outputs the data.
	`nocheck`	N	Outputs the data in the database as is.
No	`check`	Y*	Outputs the data in the database as is.
	`replace`		
	`nocheck`	N	

Legend:

Y: Checks for control characters

N: Does not check for control characters

* Checks for 1-byte characters only; does not check for multi-byte characters.

(b) enclose=enclosing-character

When a column data item is to be enclosed with a character in order to identify the data, specifies the 1-byte enclosing character to be used. Rules for the enclosing character are the same as for the replacement character in the `cntlcode` operand.

Criteria

Specify this operand when you wish to identify a column data item by enclosing it with a particular character. When a variable-length character string (character string data, national character data, or mixed character string data type) is shorter than the length defined for the column data or is the null value, the system pads the data up to the defined length (the default uses the space character). If the length of the column data is 0 bytes, the resulting data cannot be identified from 0-byte character string data, null value data, or data consisting of only space characters. By specifying the `enclose` operand, the column data in such cases can be identified because it will be enclosed with the specified enclosing character.

The following table shows output examples when the data type is `varchar(8)` and the padding character is the space character:

Data	Output data	
	enclose operand omitted	**enclose="**
0-byte characters	△ △ △ △ △ △ △ △	"" △ △ △ △ △ △ △ △
Null value	△ △ △ △ △ △ △ △	△ △ △ △ △ △ △ △ △ △
All space characters	△ △ △ △ △ △ △ △	" △ △ △ △ △ △ △ △ "

Legend:

△ : One space character

Rules for the enclose operand

1. Only one `enclose` operand can be specified in the `fixtext_option` statement.
2. The following table describes whether or not the enclosing character is used when the `enclose` operand is specified:

Data type of column	Data value in column	Whether or not enclosing character is used
`CHAR`, `VARCHAR`, `NCHAR`, `NVARCHAR`, `MCHAR`, and `MVARCHR`	Null value	Not used (padding character is used)*
	Real value	Used*
Other	Null value	Not used
	Real value	

* The output length increases by 2 bytes.

(c) format=data-type,output-format

Specifies that the output format for column data is to be changed. The supported data types are `INTEGER` and `SMALLINT`.

data-type:

Specifies the data type of the column or parameter (constructor parameter reverse creation function) whose output format is to be changed. Specify either `INTEGER` or `SMALLINT`.

Rules

1. The same data type cannot be specified more than once (value of

data-type).

2. If the table does not contain the specified data type, the specification is ignored.

output-format:

Specifies the output format for the data type, as either `type1` or `type2`.

When this operand is omitted, `type1` is assumed. If the data type is neither `INTEGER` nor `SMALLINT`, `type 1` is assumed.

For details about `type1` and `type2`, see *8.3.3(3)(a) Output format*.

8.10 Reorganization using a UOC

If you use a UOC, you can pass data retrieved from the database to the UOC, use the UOC to edit the data, and then output the data to an unload data file.

8.10.1 How to use a UOC during reorganization

You use the reorganization method that uses a UOC mainly in the following cases:

- Deleting unneeded data

  Use a UOC for reorganization when you want to reorganize a table and delete unneeded data at the same time or when you want to reload an unload data file from which unneeded data has been deleted to another table.

- Updating data for use by applications

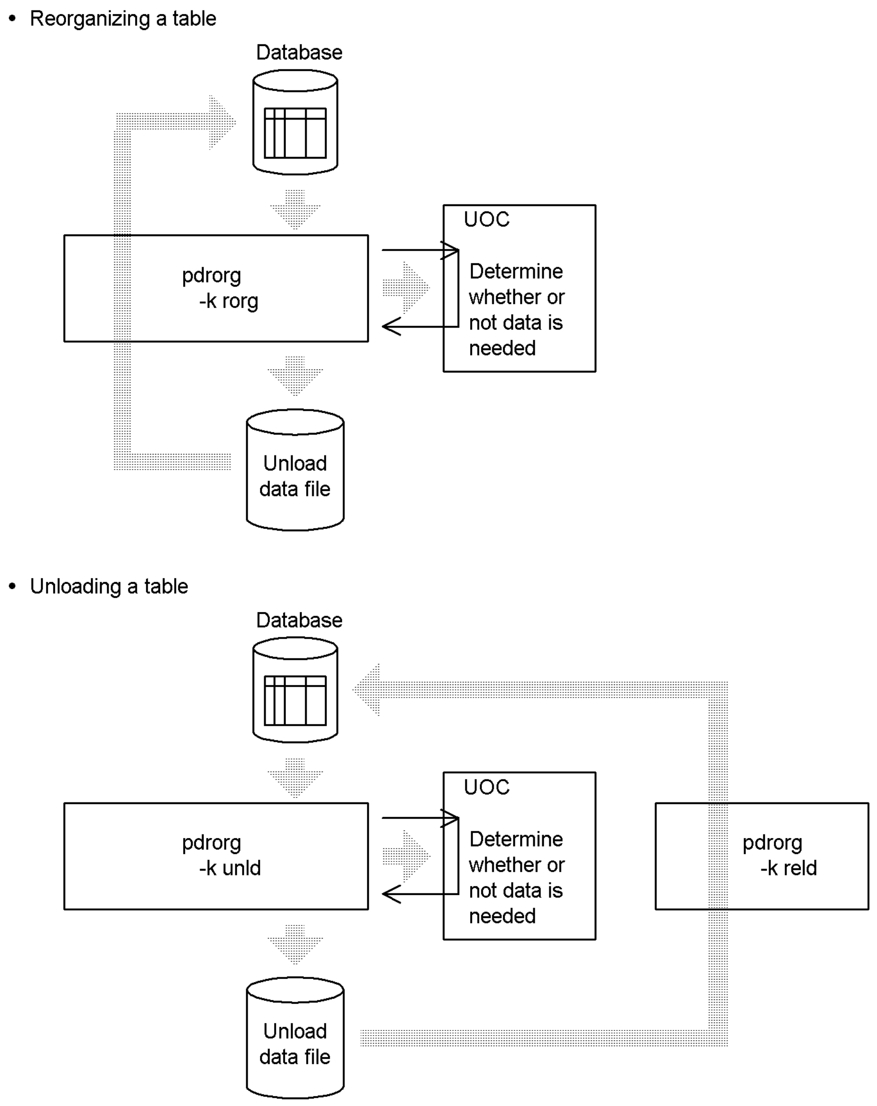
  Use a UOC for reorganization when you want to update data to create an input data file for `pdload` or you want to output data in a desired format for use by applications.

(1) Deleting unneeded data

You can output only data that is needed to an unload data file by using a UOC during the unload operation for table reorganization or during the table unload operation to determine which data is needed.

Figure 8-20 provides an overview of reorganization using a UOC (deleting unneeded data).

Figure 8-20: Overview of reorganization using a UOC (deleting unneeded data)



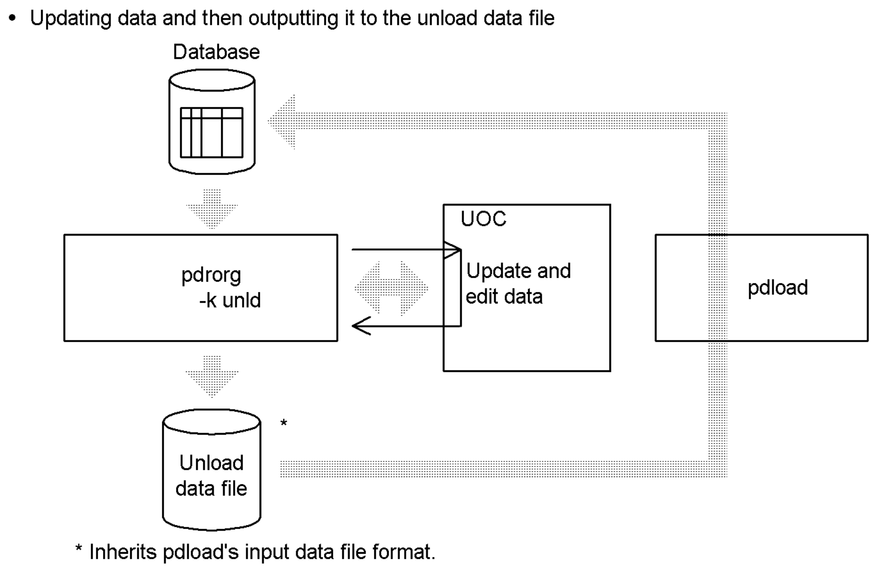
(2) Updating data for use by applications

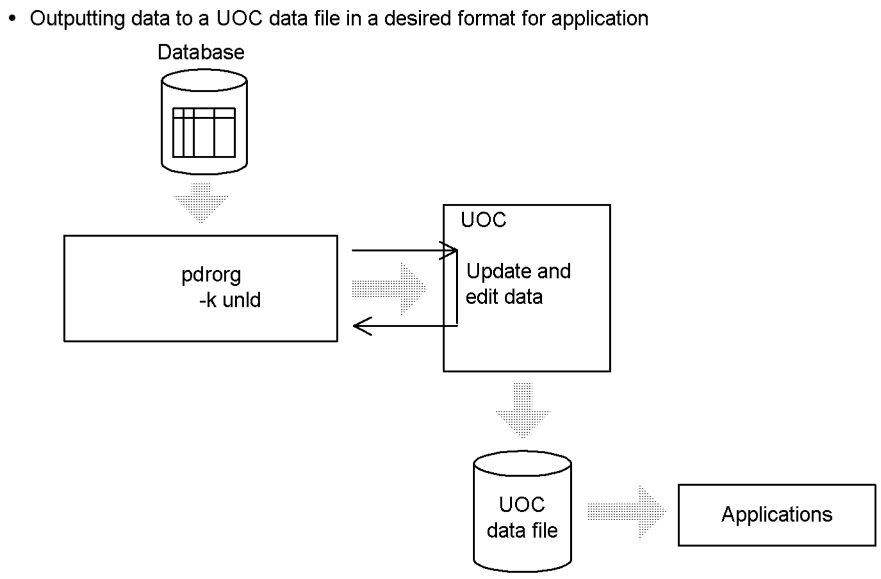
When you unload a table, you can use a UOC to update or edit the data and output it to an unload data file. The resulting unload data file is in `pdload`'s input data file format.

You can also use a UOC to update or edit data and save it in a file in a desired format. The obtained file is called a *UOC data file*.

Figure 8-21 provides an overview of reorganization using a UOC (updating data for use by applications).

Figure 8-21: Overview of reorganization using a UOC (updating data for use by applications)


• Updating data and then outputting it to the unload data file
Database
pdrorg
-k unld
UOC
Update and edit data
pdload
Unload data file
*
* Inherits pdload's input data file format.


• Outputting data to a UOC data file in a desired format for application
Database
pdrorg
-k unld
UOC
Update and edit data
UOC data file
Applications

Note that you can update data for use by applications only when `-k unld` is specified, in which case you also need to specify the `-W` option.

- Checking post-update data by `pdrorg`

  `pdrorg` checks the data values set in the post-update data address list in the UOC interface area that is returned from the UOC. If `pdrorg` finds any invalid data, it cancels the processing and terminates with return code `8`.

  Table 8-21 lists the check items for each data type in the post-update data.

  Table 8-21: Check items for each data type in the post-update data

Data type	**Check item**				
	Length of variable-length data[*1]	**Sign**[*2]	**Data range**[*3]	**Repetition column**	
				Number of elements	**Null value**
`INTEGER`, `SMALLINT`, `FLOAT`, `SMALLFLT`	—	N	N	Y	Y
`DECIMAL`	—	Y	Y	Y	Y
`CHAR`, `MCHAR`, `NCHAR`	—	—	N	Y	Y
`DATE`, `TIME`	—	—	Y[*4]	Y	Y
`INTERVAL YEAR TO DAY`, `INTERVAL HOUR TO SECOND`	—	Y	Y[*4]	Y	Y
`VARCHAR`, `MVARCHAR`, `NVARCHAR`	Y	—	N	Y	Y
`TIMESTAMP`	—	—	Y[*4]	Y	Y
`BINARY`	Y	—	N	—	—
`BLOB` (`BLOB` attribute)	Y	—	N	—	—
Abstract data type (parameter value)	Depends on the data type of each parameter.				

Legend:

Y: Checked.

N: Not checked.

— : Not applicable

[*1] Checks the length part of the variable-length character string, `BINARY`, or large object data type. If the value of the length part is negative, an error results.

In the case of a variable-length character string, an error results if the value of the length part exceeds 32,000 bytes.

If a column has the large object data type or the return value of the constructor parameter reverse creation function specified in the `unld_func` statement has the large object data type, `bloblimit` is specified in the `unlduoc` statement, and the data exceeds the specified memory space allocation size, an error results. If `bloblimit` is not specified in the `unlduoc` statement, an error results if the data size exceeds 2 gigabytes.

[*2] Checks the sign part (4 bits) in the packed format. An error results if the sign part is not `0xC`, `0xD`, or `0xF`.

[*3] For the packed format, an error results if the packed data (4 bits) is none of `0x0` to `0x9`. For other formats, `pdrorg` does not check the contents of data values.

[*4] An error results if the value is invalid as date, time, or time stamp data (such as the month part of date data exceeding 12).

8.10.2 Relationships between options and control statements

(1) Options and control statements that are specified during reorganization using a UOC

Table 8-22 shows the options and control statements that are specified during reorganization using a UOC.

Table 8-22: Options and control statements that are specified during reorganization using UOC

Type			Option		Control statement		Output to file	
			-k	-W	unload statement	unlduoc statement	pdrorg	UOC
Deleting unneeded data[*1]	Table reorganization		`rorg`	N	Y[*3]	Y	Output	Optional
	Table unload + reload operation	Unload operation	`unld`	—	Y[*3]	Y	Output	Optional
		Reload operation	`reld`	—	Y[*3]	—	—	—
Updating data for use by applications[*2]	File output by `pdrorg`		`unld`	Y	Y[*3]	Y	Output	Optional
	File output by UOC		`unld`	N	Y[*4]	Y	Not applica ble	Require d

Legend:

Y: Specify.

N: Cannot be specified (results in an error if specified).

— : Ignored, if specified.

*1 If the table contains columns of an abstract data type, data is passed to the UOC only for the abstract data type columns for which the constructor parameter reverse creation function has been specified in the `unld_func` statement. In this case, the null value is passed to the UOC, but the retrieved data is stored in the unload data file. Therefore, the UOC cannot determine whether or not the data is needed.

*2 If the table contains columns of an abstract data type, specify the `unld_func` statement for all such abstract data type columns. If the `unld_func` statement is not specified, an error results.

*3 Specify the name of the unload data file in the `unload` statement.

*4 Specify `(uoc)` in the unload statement.

(2) Option specification rules for reorganization using a UOC

The following are the option specification rules for reorganization using a UOC:

1. You can use a UOC for reorganization during table reorganization (`-k rorg`) or a table unload operation (`-k unld`).
2. If the table contains a column of large object data type or an abstract data type with the large object data attribute, you must specify the `-j` option. An error results if the `-j` option is omitted.
3. `dic` (data dictionary table is to be processed) cannot be specified in the `-c` option. Specifying `dic` results in an error.
4. If you specify `(uoc)` in the `unload` statement, you cannot specify the `-W` option. Specifying the `-W` option results in an error.
5. If you specify `(uoc)` in the `unload` statement, specifying the `-f` option results in an error.

(3) Control statement specification rules for reorganization using a UOC

The following are the control statement specification rules for reorganization using a UOC:

1. To use a UOC for reorganization, you must always specify the following two control statements:
   - `unlduoc` statement
   - `unload` statement
2. The `lobunld` statement cannot be specified.

(4) Relationship between the host and the server that calls the UOC

Table 8-23 describes the relationship between the host and the server that calls the

UOC. You must provide the shared library that contains the applicable UOC on the appropriate host, as indicated in Table 8-23.

Table 8-23: Relationship between the host and the server that calls the UOC

Execution environment				Server	Host
HiRDB/Single Server	`-g` omitted			Single server	Host where the single server is located
	`-g` specified			Utility server (`pdrorgm`)	Host specified in the `unload` statement
HiRDB/Parallel Server	`-g` omitted	In units of tables	Non-partitioned table	Back-end server	Host where the server containing the table is located
			Partitioned table		All hosts where the servers specified in the `unload` statement are located
		In units of RDAREAs			Host where the server containing the specified RDAREA is located
	`-g` specified			Utility server (`pdrorgm`)	Host specified in the `unload` statement

(5) Data that is passed to the UOC and pdrorg's data conversion timing

`pdrorg` provides a function for converting the data according to specified options and control statements. Reorganization using a UOC can also use this function. This data conversion function passes unconverted data to the UOC and converts data before saving the data in the unload data file. Table 8-24 describes the data to be passed to the UOC and `pdrorg`'s data conversion timing.

Table 8-24: Data that is passed to the UOC and pdrorg's data conversion timing

Type	Option or control statement available for data conversion	Data passed to UOC	Data conversion timing
Deleting unneeded data (table reorganization)	`-S`	Data with spaces	When unload data file is output
	`-W`	—	—
	`-W dat,sup` or `-W extdat,sup`	—	—
	`spacelvl` in `option` statement	Data without space conversion	When table is reloaded
	`blobtovarchar` statement	—	—
Deleting unneeded data (table unload + reload operations)	`-S`	Data with spaces	When unload data file is output
	`-W`	—	—
	`-W dat,sup` or `-W extdat,sup`	—	—
	`spacelvl` in `option` statement	Data without space conversion	When table is reloaded
	`blobtovarchar` statement	—	—
Updating data for use by applications (file output by `pdrorg`)	`-S`	—	—
	`-W`	Data before being converted to DAT or binary format	When unload data file is output*
	`-W dat,sup` or `-W extdat,sup`	Data with spaces	When unload data file is output*
	`spacelvl` in `option` statement	Data without space conversion	When unload data file is output*
	`blobtovarchar` statement	Data in the BLOB format	When unload data file is output*

Type	Option or control statement available for data conversion	Data passed to UOC	Data conversion timing
Updating data for use by applications (file output by UOC)	`-S`	Data with spaces	—
	`-W`	—	—
	`-W dat,sup` or `-W extdat,sup`	—	—
	`spacelvl` in `option` statement	Data without space conversion	—
	`blobtovarchar` statement	—	—

Legend:

—: Not applicable

Note

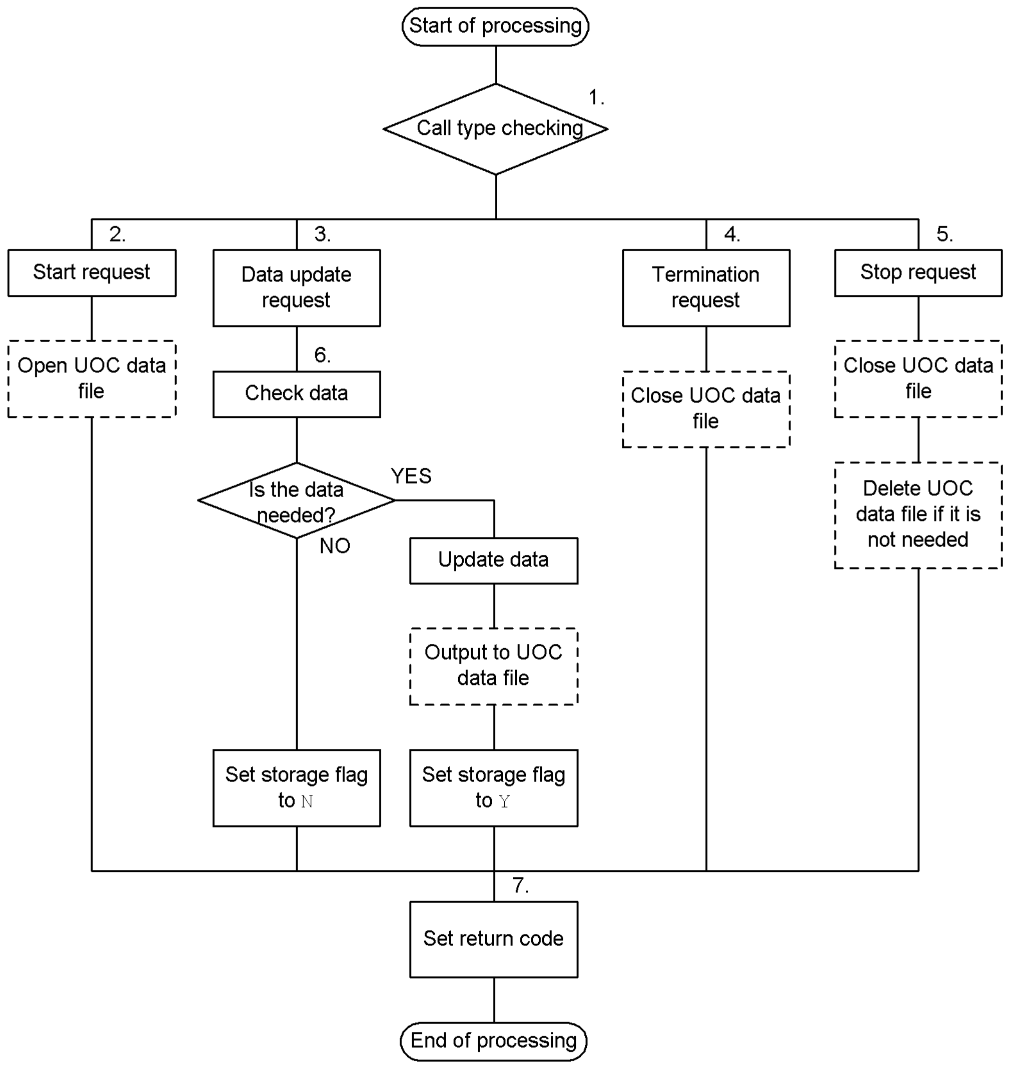
If the table contains a column of `DECIMAL` type for which `SUPPRESS` was specified in the table definition, the data obtained after expansion is passed to the UOC. The data is converted when the table is reloaded.

* The data obtained after editing by the UOC is converted.

8.10.3 Overview of UOC processing

Figure 8-22 provides an overview of UOC processing and Figure 8-23 shows the call sequence between `pdrorg` and UOC.

Figure 8-22: Overview of UOC processing



Note: Execute the processing enclosed in the dashed line when you use a UOC to create a UOC data file.

Explanation:

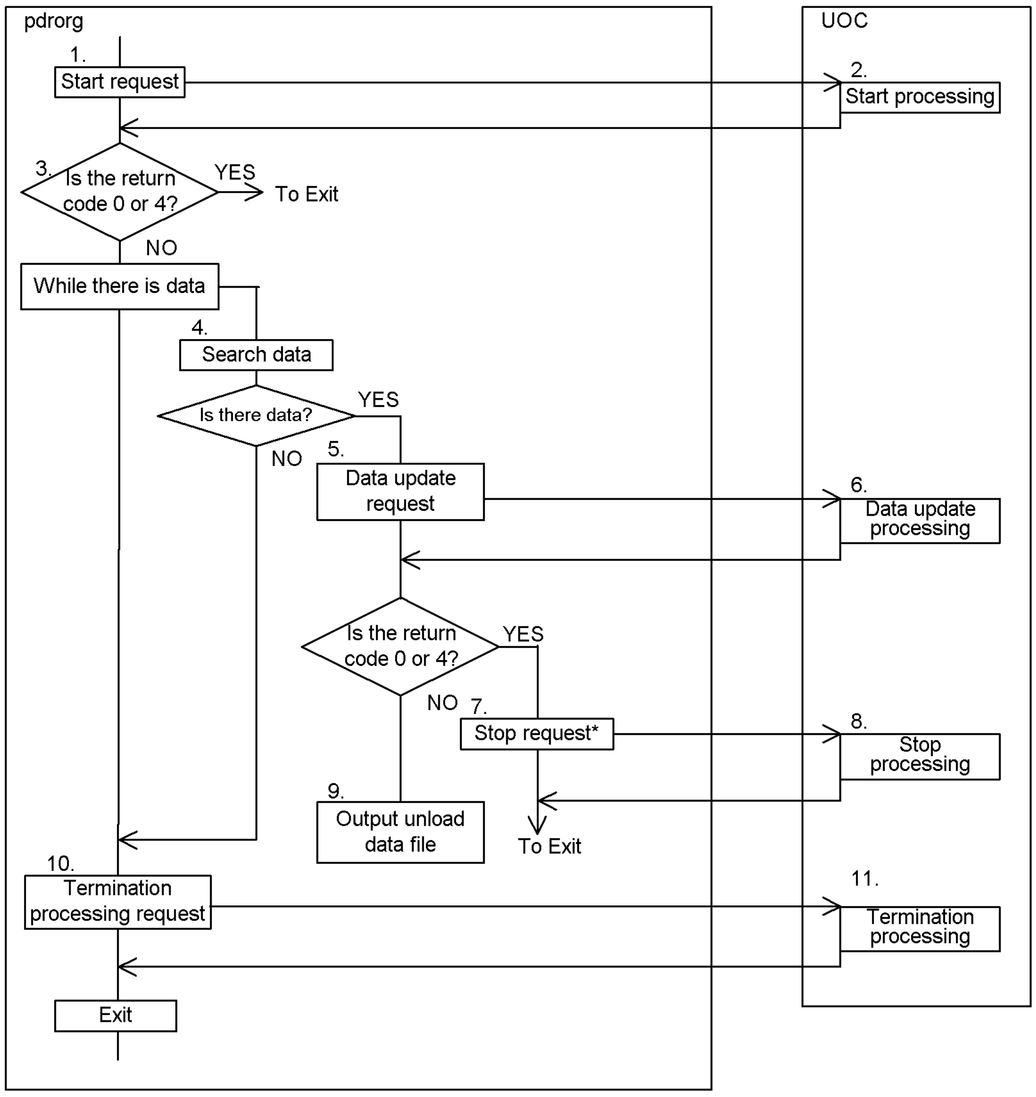
Control is passed from `pdrorg` to the UOC and processing is started. When the processing is finished, control is returned to `pdrorg`.

1. Checks the call type set by `pdrorg` in the UOC interface area.
2. Performs start processing, such as initialization and UOC data file open

processing.

3. Updates or edits the data set by `pdrorg` in the UOC interface area.
4. When there is no more data, the UOC performs termination processing, such as closing the UOC data files.
5. If `pdrorg` processing or internal UOC processing results in an error, the UOC performs termination processing, such as closing the UOC data files and deleting unneeded UOC data files.
6. Checks the row data to determine whether or not it is needed. If the row data is needed, the UOC updates or edits the data and sets the storage flag in the UOC interface area to `Y`. If the row data is not needed, the UOC sets the storage flag in the UOC interface area to `N`. If the UOC is to create a UOC data file, it saves the updated or edited row data in the UOC data file.
7. Sets the return code in the UOC interface area.

Figure 8-23: Call sequence between pdrorg and UOC



* If any of the `pdrorg` processing 3-9 results in an error, the termination processing request in 7 is executed immediately. When control is returned from the UOC, `pdrorg` cancels processing and terminates itself.

Explanation:

In the case of reorganization or an unload operation in units of schemas, steps 1-11 are repeated for each table in the schema. If an error occurs during table processing, `pdrorg` processing is cancelled immediately.

1. `pdrorg` sets the start request as the call type and calls the UOC.
2. If the type of call from `pdrorg` is a start request, the UOC executes the following processing:
   - If the UOC is to output a UOC data file, it creates (opens) the UOC data file.
   - The UOC performs preparations for starting processing, such as allocation of various areas.
   - If the preparations are completed successfully, the UOC sets return code `0` in the UOC interface area; if not, the UOC sets return code `8` and returns control.

   If return code `8` is set, you must perform post-processing, such as closing the UOC data file because the UOC can no longer be called (no stop request is issued).
3. `pdrorg` checks the UOC's return code set in the UOC interface area. If it is neither `0` nor `4`, `pdrorg` cancels its processing immediately.
4. `pdrorg` retrieves data in the database.
5. If `pdrorg` was able to retrieve table data, it set the retrieved data in the data address list, sets a data update request as the call type, and calls the UOC. When there is no more data to be retrieved, `pdrorg` performs step 10 (termination processing).

   The data update request is processed repeatedly until any of the following conditions is satisfied:
   - There is no more table data.
   - The UOC sets a return code other than `0` or `4` in the UOC interface area.
   - An internal processing error occurs in `pdrorg`.
6. If the type of call from `pdrorg` is a data update request, the UOC executes the following processing:

   If each process terminates normally, UOC sets return code `0` in the UOC interface area; if a process terminates abnormally, the UOC sets return code `8` and returns control.

   (a) When deleting unneeded data for table reorganization or the unload operation

   The UOC references the data that was passed from `pdrorg` to the data address list in the UOC interface area. If the data is needed, the UOC sets the storage flag to `Y` in the UOC interface area. If the data is not needed, the UOC sets the storage flag to `N`.

   (b) When updating data for use by applications

   The UOC executes processing such as data updating.

If the UOC has updated data, it sets the post-update data address list in the UOC interface area and sets the storage flags to `Y`.

If the data is unneeded, the UOC sets the storage flag to `N` in the UOC interface area.

If the UOC is to output a UOC data file, it saves the needed data in the UOC data file.

7. If the UOC has set return code `8` in the UOC interface area, `pdrorg` sets a stop request as the type of call and calls the UOC.
8. If the type of call from `pdrorg` is a stop request, the UOC executes termination processing. After this call, `pdrorg` cannot call the UOC again.
    - If you have created a UOC data file, close it. If you no longer need the created UOC data file, delete it at this stage.
    - Perform post-processing, such as freeing the allocated memory.
9. If `pdrorg` is to output an unload data file and the storage flag in the UOC interface area is set to `Y`, `pdrorg` outputs data to the unload data file. If the storage flag is set to `N`, `pdrorg` does not output the data. If the data is to be updated (`-W` option is specified), `pdrorg` references the post-update data address list in the UOC interface area and outputs the data updated by the UOC to the unload data file in the format specified in the `-W` option.
10. If all the following conditions are satisfied, `pdrorg` sets a termination request as the type of call and calls the UOC:
    - All data has been retrieved from the specified table.
    - An internal error occurred in `pdrorg`.
    - The return codes set by the UOC in the UOC interface area are all normal (`0` or `4`).
11. If the type of call from `pdrorg` is a termination request, the UOC executes termination processing.
    - If you have created a UOC data file, close it.
    - Perform post-processing, such as freeing the allocated memory.

8.10.4 UOC interface

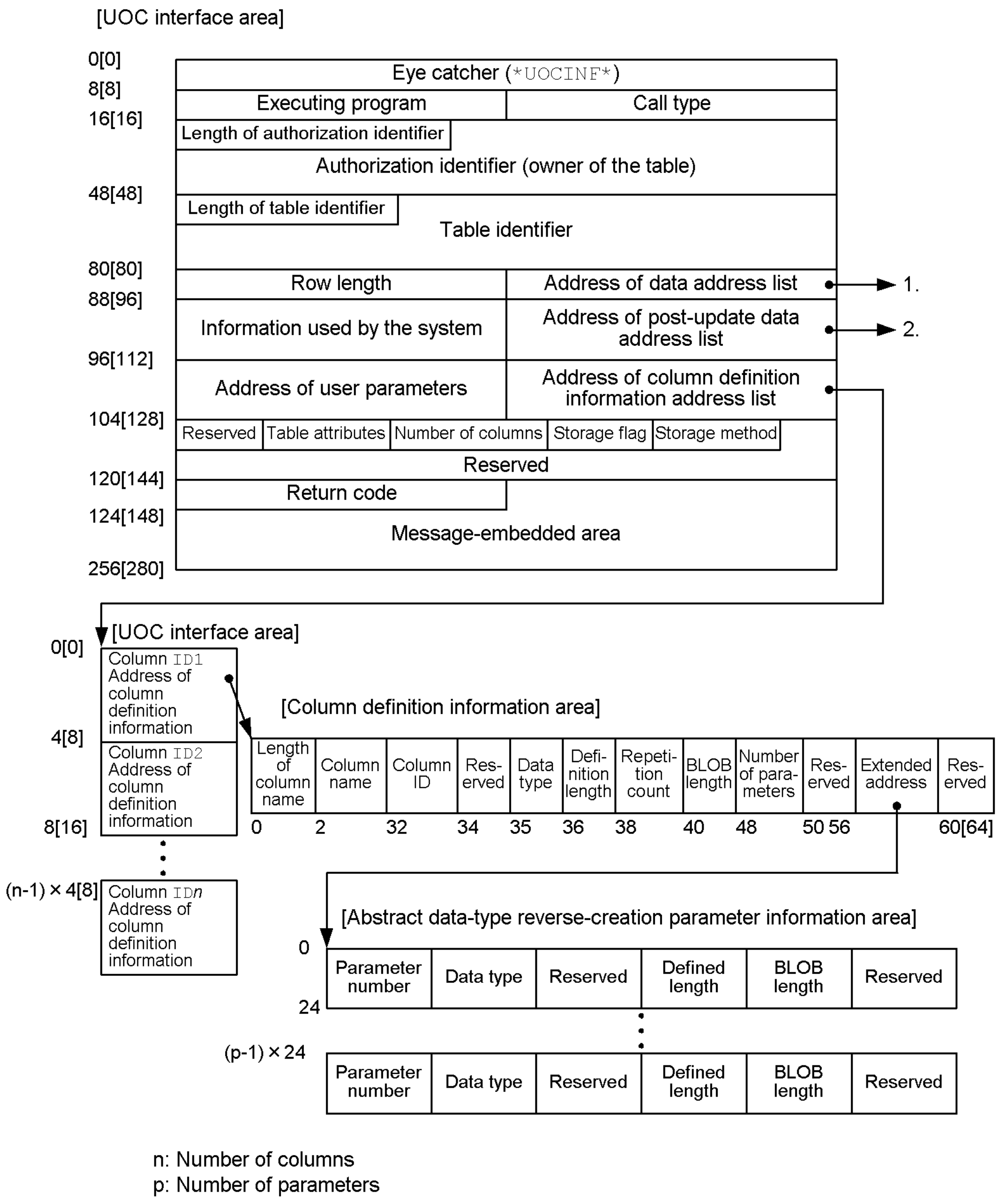
(1) Structure and contents of the UOC interface area

An interface area is used to exchange information between `pdrorg` and a UOC. It is called a *UOC interface area*.

`pdrorg` always allocates this area except for the update buffer. The UOC receives the address of this area in the first argument of a call function to reference and update information.

Figures 8-24 and 8-25 show the structure of the UOC interface area.

Figure 8-24: Structure of the UOC interface area (1/2)



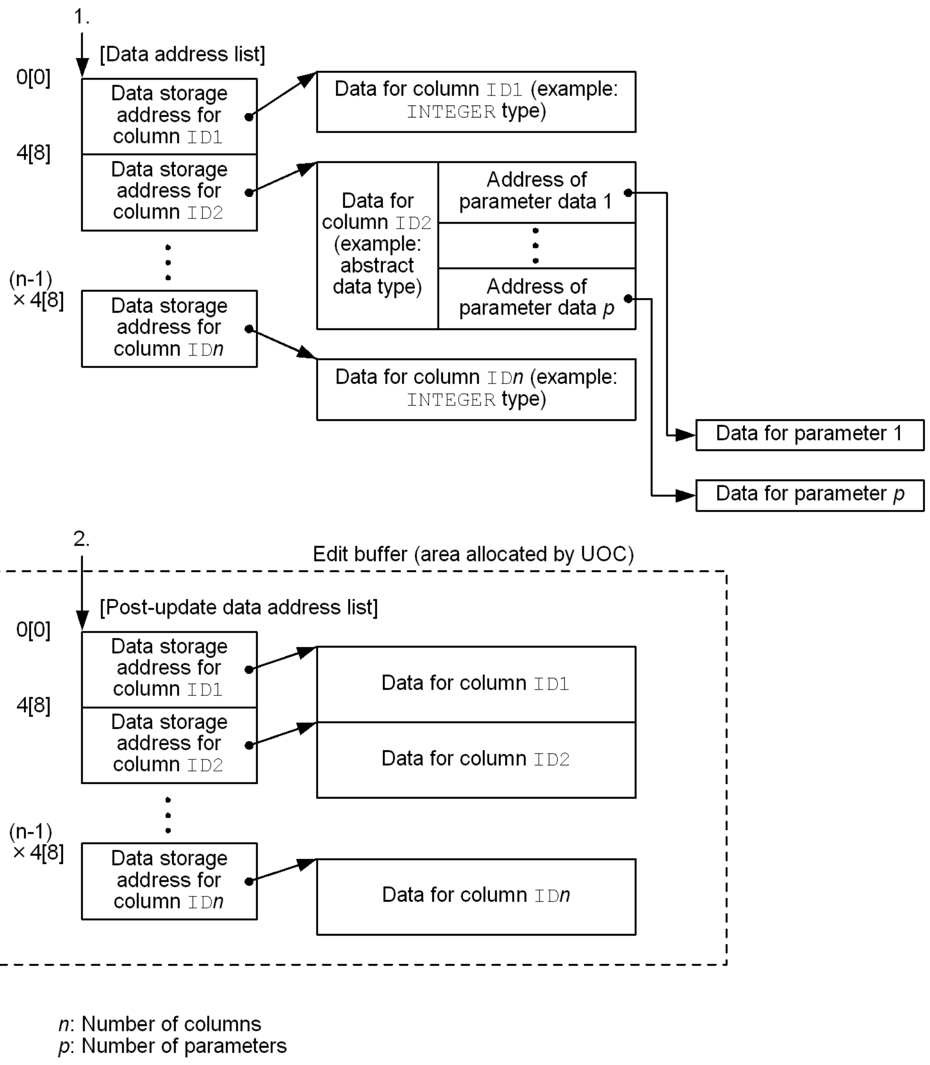
Note 1

Square brackets [ ] enclose the numeric value for HiRDB in the 64-bit mode.

Note 2

For details about the contents of the UOC interface are, see Table 8-25; for details about the contents of the column definition information address list, see Table 8-26; for details about the contents of the column definition information area, see Table 8-27; for details about the contents of the abstract data-type reverse-creation parameter information area, see Table 8-28.

Figure 8-25: Structure of UOC interface area (2/2)


1.
[Data address list]
0[0]
4[8]
(n-1) × 4[8]
Data storage address for column ID1
Data storage address for column ID2
Data storage address for column IDn
Data for column ID1 (example: INTEGER type)
Data for column ID2 (example: abstract data type)
Address of parameter data 1
Address of parameter data p
Data for column IDn (example: INTEGER type)
Data for parameter 1
Data for parameter p
2.
Edit buffer (area allocated by UOC)
[Post-update data address list]
0[0]
4[8]
(n-1) × 4[8]
Data storage address for column ID1
Data storage address for column ID2
Data storage address for column IDn
Data for column ID1
Data for column ID2
Data for column IDn
n: Number of columns
p: Number of parameters

Note 1

Square brackets [ ] enclose the numeric value for a HiRDB in the 64-bit mode.

Note 2

For details about the contents of the data address list, see Table 8-31; for details about the contents of the post-update data address list, see Table 8-32.

Table 8-25: Contents of the UOC interface area

Relative location		Field name	Length (bytes)		Attribute	Program setting the value	Description
32	64		32	64			
0	0	Eye-catcher	8	8	`char`	`pdrorg`	Interface area (`'*UOCINF*'`)
8	8	Running program	4	4	`int`	`pdrorg`	Program that has control: `0`: `pdrorg` `1`: UOC
12	12	Call type	4	4	`int`	`pdrorg`	Type of processing request to the UOC: `o`: Start request `e`: Data update request `c`: Termination request `t`: Stop request
16	16	Length of authorization identifier	2	2	`short`	`pdrorg`	Length of the target table's owner name
18	18	Authorization identifier	30	30	`char`	`pdrorg`	Target table's owner name
48	48	Length of table identifier	2	2	`short`	`pdrorg`	Length of the target table name
50	50	Table identifier	30	30	`char`	pdrorg	Name of the target table
80	80	Row length	4	8	`long`	`pdrorg`	If the data storage method is `Y`, this field sets the row length.
84	88	Address of the data address list	4	8	`void*`	`pdrorg`	Start address of the data address list that contains the address of the data retrieved by `pdrorg`

Relative location		Field name	Length (bytes)		Attribute	Program setting the value	Description
32	64		32	64			
88	96	Information used by the system	4	8	`long`	`pdrorg`	Information used by the system (UOC may not use this information)
92	104	Address of the post-update data address list	4	8	`void*`	`UOC`	If the data updated by the UOC is to be returned to `pdrorg`, this field sets the start address of the post-update data address list that contains the address of the data created within the UOC. If the data is not to be updated, this field sets `0`.
96	112	Address of user parameter	4	8	`void*`	`pdrorg`	Address of the character string that was specified in the `param` operand of the `unlduoc` statement (the character string ends with `/0`). If the `param` operand was omitted, the null value is set.
100	120	Address of the column definition information address list	4	8	`void*`	`pdrorg`	Start address of the column definition information address list for the target table
104	128	Reserved	1	1	`char`	`pdrorg`	Reserved area (UOC may not use this information)
105	129	Table attribute	1	1	`char`	`pdrorg`	Attribute of the target table: `F`: FIX table blank: Non-FIX table
106	130	Number of columns	2	2	`short`	`pdrorg`	Number of columns in the target table
108	132	Storage flag	1	1	`char`	UOC	Storage flag: `Y`: Stores data in unload data file. `N`: Does not store data in unload data file. If `pdrorg` is used to create an unload data file, make sure that this flag is set (if the UOC is used to create a UOC data file, there is no need to set this flag).

Relative location		Field name	Length (bytes)		Attribute	Program setting the value	Description
32	64		32	64			
109	133	Data storage method	1	1	`char`	`pdrorg`	For a FIX table, this field sets the value of the `fixrow` operand specified in the `unlduoc` statement (when `fixrow` is omitted, the field sets `N`). `Y`: Passes data in the order the columns were defined (consecutively). `N`: Passes the start address of the data that has been corrected according to the boundary of data type.
110	134	Reserved	10	10	`char`	`pdrorg`	Reserved area (UOC may not use this information)
120	144	Return code	4	4	`int`	UOC	Return code. The UOC that has received control from `pdrorg` must set a return code for each call according to the following guidelines: `0`: Set this code if the processing for each request was executed normally. `4`: Set this code if you want to display a message even when processing was normal, such as for debugging. The meaning is the same as for `0`. When the return code is set to `4`, the message is displayed up to three times. Note that once the message has been displayed three times, the utility ignores return code `4`, if set again, in which case there is no more message display.

Relative location		Field name	Length (bytes)		Attribute	Program setting the value	Description
32	64		32	64			
							`8`: Set this code if an error occurs during UOC processing. If the set return code is none of the above, `pdrorg` cancels processing. When the return code is `4` or `8`, `pdrorg` outputs the contents of the message-embedded area to the standard output and message log. Make sure that the message consists of no more than 131 bytes of a character string that ends with `\0`. `pdrorg` does not output any message that begins with `\0`.
124	148	Message-embedded area	132	132	`char`	`pdrorg` UOC	Storage area for the message that is output to the standard output and message log (`pdrorg` places `\0` at the beginning of the message before passing it).

Legend:

32: Shows the relative location or length for a HiRDB in the 32-bit mode.

64: Shows the relative location or length for a HiRDB in the 64-bit mode.

Note 1

For the start address, `pdrorg` guarantees a 4-byte boundary for a HiRDB in the 32-bit mode and an 8-byte boundary for a HiRDB in the 64-bit mode.

Note 2

The following table describes whether or not each field can be referenced and updated:

Field name	Call type			
	Start request	Data update request	Termination request	Stop request
Eye-catcher	R	R	R	R
Running program	R	R	R	R
Call type	R	R	R	R

Field name	Call type			
	Start request	Data update request	Termination request	Stop request
Length of authorization identifier	R	R	R	R
Authorization identifier	R	R	R	R
Length of table identifier	R	R	R	R
Table identifier	R	R	R	R
Row length	N	R	N	N
Address of the data address list	N	R	N	N
Address of the post-update data address list	N	Y	N	N
Address of user parameter	R	R	R	R
Address of the column definition information address list	R	R	R	R
Table attribute	R	R	R	R
Number of columns	R	R	R	R
Data storage method	N	Y	N	N
Message-embedded area	Y	Y	Y	Y
Return code	Y	Y	Y	Y
Storage flag	N	Y	N	N

Legend:

Y: Value can be set.

R: Value can be referenced.

N: Value cannot be referenced (value is not guaranteed).

Table 8-26: Contents of column definition information address list

Relative location		**Field name**	**Length (bytes)**		**Attribute**	**Program setting the value**	**Description**
32	**64**		**32**	**64**			
0	0	Definition information address for column `ID1`	4	8	`void*`	`pdrorg`	Sets the address of the column definition information for column `ID1`.
4	8	Definition information address for column `ID2`	4	8	`void*`	`pdrorg`	Sets the address of the column definition information for column `ID2`.
(*n* - 1) × 4	(*n* - 1) × 8	Definition information address for column `ID`*n*	4	8	`void*`	`pdrorg`	Sets the address of the column definition information for column `ID`*n*.

Legend:

32: Shows the relative location or length for a HiRDB in the 32-bit mode.

64: Shows the relative location or length for a HiRDB in the 64-bit mode.

Note 1

For the start address, `pdrorg` guarantees a 4-byte boundary for a HiRDB in the 32-bit mode and an 8-byte boundary for a HiRDB in the 64-bit mode.

Note 2

The contents of the column definition information address list are passed consecutively in the order of the column definitions.

Table 8-27: Contents of column definition information area

Relative location		**Field name**	**Length (bytes)**		**Attribute**	**Program setting the value**	**Description**
32	**64**		**32**	**64**			
0	0	Length of column name	2	2	`short`	`pdrorg`	Sets the length of the column name.
2	2	Column name	30	30	`char`	`pdrorg`	Sets the column name.
32	32	Column ID	2	2	`short`	`pdrorg`	Sets the column ID.

Relative location		Field name	Length (bytes)		Attribute	Program setting the value	Description
32	64		32	64			
34	34	Reserved 1	1	1	—	—	—
35	35	Data type	1	1	`unsigned char`	`pdrorg`	Sets the data type of the column. For details about the data code and data value boundary for each data type, see Table 8-30.
36	36	Defined length	2	2	`short`	`pdrorg`	Sets the defined length of the column. For details about the length and contents of the definition length area for columns, see Table 8-29.
38	38	Repetition count	2	2	`short`	`pdrorg`	Sets the repetition count for the column.
40	40	`BLOB` or `BINARY` length	8	8	`int[2]`	`pdrorg`	This field is set if the column's data type is large object data or `BINARY`. The first 4 bytes are set to 0 and the remaining 4 bytes contain the defined length of large object data type in bytes.
48	48	Number of parameter	2	2	`short`	`pdrorg`	For an abstract data type, this field sets the number of functions in the corresponding `unld_func` statements (number of parameters). A value of `0` is set in the following cases: • The column's data type is not an abstract data type. • No `unld_func` statement is specified for the abstract data type.
50	50	Reserved 2	6	6	`char`	`pdrorg`	Reserved area
56	56	Extended address	4	8	`void*`	`pdrorg`	Sets the start address of the abstract data type reverse-creation parameter information. A value of `0` is set if the number of parameters is 0.
60	64	Reserved 3	64	64	`char`	`pdrorg`	Reserved area

Legend:

32: Shows the relative location or length for a HiRDB in the 32-bit mode.

64: Shows the relative location or length for a HiRDB in the 64-bit mode.

Note

For the start address, `pdrorg` guarantees a 4-byte boundary for a HiRDB in the 32-bit mode and an 8-byte boundary for a HiRDB in the 64-bit mode.

Table 8-28: Contents of the abstract data type reverse-creation parameter information area

Relative location	Field name	Length (bytes)	Attribute	Program setting the value	Description
0	Parameter number	2	`short`	`pdrorg`	Parameter number (beginning with 1)
2	Data type	1	`unsigned char`	`pdrorg`	Data type of the parameter. For details about the data code and data value boundary for each data type, see Table 8-30.
3	Reserved 1	3	`char`	`pdrorg`	Reserved area
6	Defined length	2	`short`	`pdrorg`	Length defined for the parameter. For details about the length and contents of the definition length area for columns, see Table 8-29.
8	`BLOB` or `BINARY` length	8	`int[2]`	`pdrorg`	If the parameter's data type is large object data or `BINARY`, this field sets the `BLOB` length. The first 4 bytes are set to 0 and the remaining 4 bytes contain the defined length of large object data type in bytes.
16	Reserved 2	112	`char`	`pdrorg`	Reserved area

Note 1

For the start address, `pdrorg` guarantees a 4-byte boundary for a HiRDB in the 32-bit mode and an 8-byte boundary for a HiRDB in the 64-bit mode.

Note 2

The contents of the abstract data type reverse-creation parameter information area are passed consecutively in the order of functions (parameters) specified in the `unld_func` statement.

Table 8-29: Length and contents of the definition length area for columns

Data type	Area size	Description
`DECIMAL`, `LARGE DECIMAL`, `INTERVAL YEAR TO DAY`, and `INTERVAL HOUR TO SECOND`	Leading 1 byte	Precision specified in the definition
	Trailing 1 byte	Unit specified in the definition
`TIMESTAMP(`*p*`)`	2 bytes	$7 + \uparrow p/2 \uparrow$
`BLOB`, abstract data type, and `BINARY`	2 bytes	Value is not guaranteed
Other	2 bytes	Defined length

Table 8-30: Data code and data value boundary for each data type

Data type	Data code (hexadecimal)		Boundary (in bytes)
	NOT NULL constraint omitted	NOT NULL constraint specified	
`MVARCHAR`	A1	A0	2
`MCHAR`	A5	A4	--
`NVARCHAR`	B1	B0	2
`NCHAR`	B5	B4	--
`VARCHAR`	C1	C0	2
`CHAR`	C5	C4	--
`FLOAT`	E1	E0	8
`SMALLFLT`	E3	E2	4
`DECIMAL`	E5	E4	--
`INTEGER`	F1	F0	4
`SMALLINT`	F5	F4	2
`INTERVAL YEAR TO DAY`	65	64	--
`DATE`	71	70	--
`TIME`	79	78	--
`INTERVAL HOUR TO SECOND`	6F	6E	--

Data type		Data code (hexadecimal)		Boundary (in bytes)
		NOT NULL constraint omitted	NOT NULL constraint specified	
TIMESTAMP		7D	7C	—
BINARY		91	90	4
BLOB		93	92	4
Abstract data type	Address list of abstract data type parameter	83	82	4 for a HiRDB in the 32-bit mode and 8 for a HiRDB in the 64-bit mode
	Data for parameter	Depends on the data type of the parameter		

Legend:

— : Address boundary is not adjusted.

Table 8-31: Contents of data address list

Relative location		Field name	Length (bytes)		Attribute	Program setting the value	Description
32	64		32	64			
0	0	Data storage address for column ID1	4	8	void*	pdrorg	Sets the address of the data for column ID1.
4	8	Data storage address for column ID2	4	8	void*	pdrorg	Sets the address of the data for column ID2.
(*n* - 1) × 4	(*n* - 1) × 8	Data storage address for column ID*n*	4	8	void*	pdrorg	Sets the address of the data for column ID*n*.

Legend:

32: Shows the relative location or length for a HiRDB in the 32-bit mode.

64: Shows the relative location or length for a HiRDB in the 64-bit mode.

Note 1

For the start address, pdrorg guarantees a 4-byte boundary for a HiRDB in the

32-bit mode and an 8-byte boundary for a HiRDB in the 64-bit mode.

Note 2

The contents of the data address list are passed consecutively in the order defined.

Note 3

If the column value is null, `0` is set.

Note 4

In the case of a FIX table, if the data storage method is `Y` in the UOC interface area, the data buffer stores a contiguous area equivalent to the row length of the UOC information beginning at the address indicated by the data storage address for column `ID1`. Data is stored in the order of column IDs.

Note 5

If the data storage method is `N` in the UOC interface area, the start address of the data is adjusted at the specified boundary for each data type. Data is not consecutive for each column.

Note 6

If the number of parameters in the column definition information area is 0, a value of 0 is set.

Table 8-32: Contents of post-update data address list

Relative location		**Field name**	**Length (bytes)**		**Attribute**	**Program setting the value**	**Description**
32	**64**		**32**	**64**			
0	0	Data storage address for column `ID1`	4	8	`void*`	UOC	Sets the address of the data for column `ID1`.
4	8	Data storage address for column `ID2`	4	8	`void*`	UOC	Sets the address of the data for column `ID2`.
(*n* - 1) × 4	(*n* - 1) × 8	Data storage address for column `ID`*n*	4	8	`void*`	UOC	Sets the address of the data for column `ID`*n*.

Legend:

32: Shows the relative location or length for a HiRDB in the 32-bit mode.

64: Shows the relative location or length for a HiRDB in the 64-bit mode.

Note 1

For the start address, guarantee a 4-byte boundary for a HiRDB in the 32-bit mode and a 8-byte boundary for a HiRDB in the 64-bit mode.

Note 2

After data has been updated, set the data address list in consecutive areas in the order of column definitions.

Note 3

You may set the data storage address contained in the data address list as the data address value. Make sure that the data value is not updated.

Note 4

There is no need to guarantee the boundary for the data address.

Note 5

Set `0` if you are changing to null the value of a column for which the `NOT NULL` constraint was not specified in the column definition. The following table describes whether or not data can be changed, depending on the `NOT NULL` constraint specification.

Column attribute	**Data before updating**	**Data after updating**	
		Non-null value	**Null value**
`NOT NULL` constraint omitted	Null value	Y	Y
	Non-null value	Y	Y
`NOT NULL` constraint specified	Non-null value	Y	N

Legend:

Y: Can be changed.

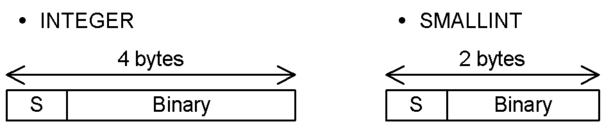
N: Cannot be changed.

(2) Format of data values

This section explains the data values of each data type.

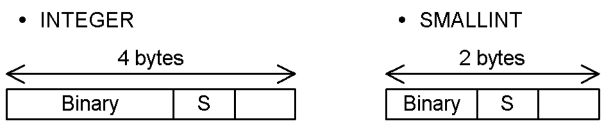
(a) INTEGER or SMALLINT

- HP-UX, Solaris, and AIX 5L



S: Sign part (1 bit):
0: Positive
1: Negative

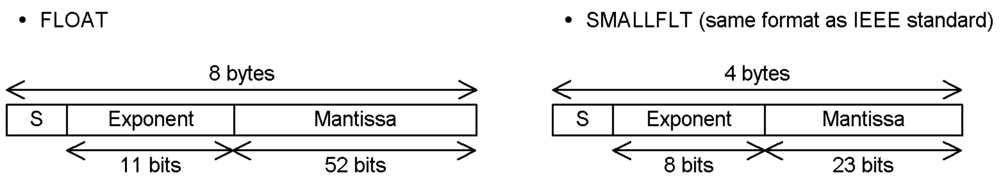
- Linux



S: Sign part (1 bit):
0: Positive
1: Negative

(b) FLOAT or SMALLFLT

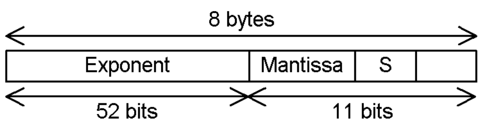
- HP-UX, Solaris, and AIX 5L



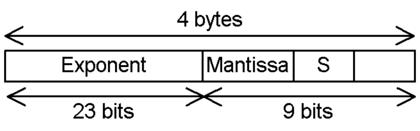
S: Sign part (1 bit):
0: Positive
1: Negative

- Linux

- FLOAT

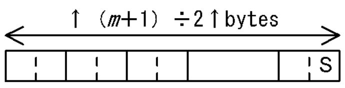


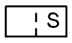
- SMALLFLT (same format as IEEE standard)



S: Sign part (1 bit)
0: Positive
1: Negative

(c) DECIMAL



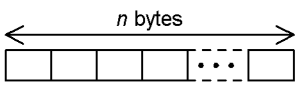


: 1 byte

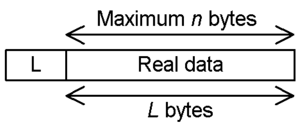
m: Precision
S: Sign part (4 bits)
Note: In the packed format, 1 byte represents 2 numeric digits.

(d) CHAR, VARCHAR, MCHAR, or MVARCHAR

- `CHAR`(*n*) or `MCHAR`(*n*)

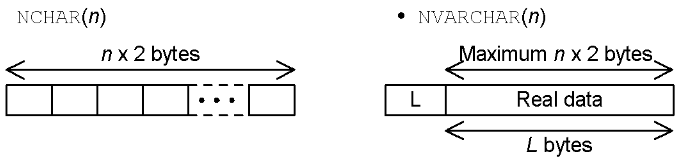


- `VARCHAR`(*n*) or `MVARCHAR`(*n*)



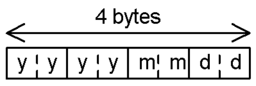
n: Defined length
L: Real data length (2 bytes)

(e) NCHAR or NVARCHAR



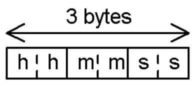
n: Defined length
L: Real data length (2 bytes)

(f) DATE



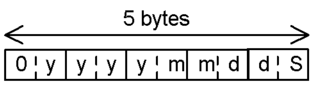
yyyy: Year
mm: Month
dd: Date
Notes:
1. DATE is a 4-byte unsigned packed decimal number.
2. In the packed format, 1 byte represents 2 numeric digits.

(g) TIME



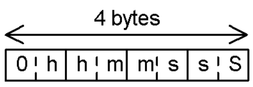
hh: Hours
mm: Minutes
ss: Seconds
Notes:
1. TIME is a 3-byte unsigned packed decimal number.
2. In the packed format, 1 byte represents 2 numeric digits.

(h) INTERVAL YEAR TO DAY



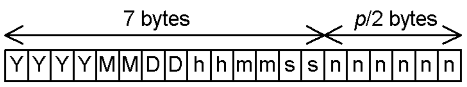
yyyy: Year
mm: Month
dd: Date
S: Signed part (4 bits)
Notes:
1. `INTERVAL YEAR TO DAY` is a 5-byte packed decimal number.
2. In the packed format, 1 byte represents 2 numeric digits.

(i) INTERVAL HOUR TO SECOND



hh: Hour
mm: Minutes
ss: Second
S: Signed part (4 bits)
Notes:
1. `INTERVAL HOUR TO SECOND` is a 4-byte packed decimal number.
2. In the packed format, 1 byte represents 2 numeric digits.

(j) TIMESTAMP

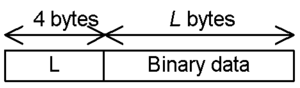


Unsigned packed decimal number with a maximum length of 10 bytes

- The fraction part is 0, 2, 4, or 6 according to the definition.
- In the packed format, 1 byte represents 2 numeric digits.

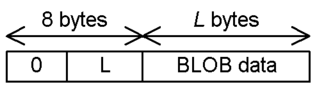
yyyy: Year
mm: Month
dd: Date
hh: Hour
mm: Minutes
ss: Second
nnnnnn: Fraction part

(k) BINARY



The first 4 bytes (*L*) contain a binary value indicating the length of data.

(l) BLOB

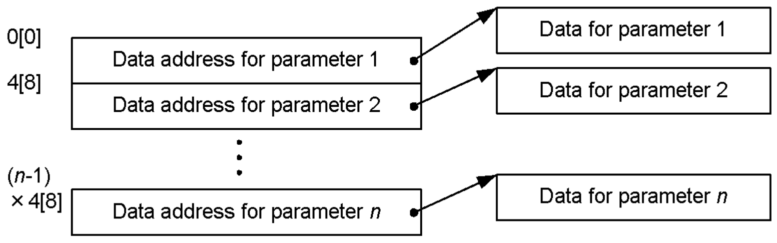


L: Data length (4 bytes)
Note:The first 4 bytes contain 0 (`0x00`) and the remaining 4 bytes contain a binary value indicating the length of the data.

(m) Abstract data type

For an abstract data type, the address list of an abstract data-type parameter is stored. Figure 8-26 shows the structure of the address list for an abstract data-type parameter, and Table 8-33 describes the contents of the address list for an abstract data-type parameter.

Figure 8-26: Structure of the address list for an abstract data-type parameter


0[0]
4[8]
(*n*-1) ×4[8]
Data address for parameter 1
Data address for parameter 2
Data address for parameter *n*
Data for parameter 1
Data for parameter 2
Data for parameter *n*

n: Number of parameters

Note

Square brackets [ ] enclose the numeric value for HiRDB in 64-bit mode.

Table 8-33: Contents of the address list for an abstract data-type parameter

Relative location		Field name	Length (bytes)		Attribute	Program setting the value	Description
32	64		32	64			
0	0	Data address for parameter 1	4	8	`void*`	`pdrorg`	Sets the address of the data for parameter 1.

Relative location		Field name	Length (bytes)		Attribute	Program setting the value	Description
32	64		32	64			
4	8	Data address for parameter 2	4	8	`void*`	`pdrorg`	Sets the address of the data for parameter 2.
(*n* - 1) × 4	(*n* - 1) × 8	Data address for parameter *n*	4	8	`void*`	`pdrorg`	Sets the address of the data for parameter *n*.

Legend:

32: Shows the relative location or length for a HiRDB in the 32-bit mode.

64: Shows the relative location or length for a HiRDB in the 64-bit mode.

Note 1

For the start address, `pdrorg` guarantees a 4-byte boundary for a HiRDB in the 32-bit mode and an 8-byte boundary for a HiRDB in the 64-bit mode.

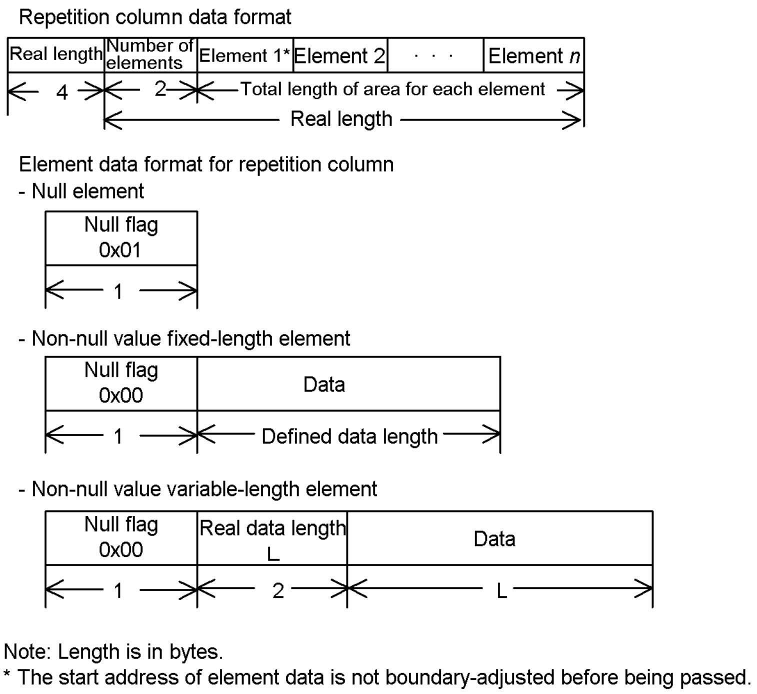
Note 2

The start address of the parameter data is adjusted by the boundary set for each data type.

Note 3

The contents of the address list for an abstract data-type parameter are passed consecutively in the order of functions (parameters) specified in the `unld_func` statement.

(n) Repetition columns


Repetition column data format
Real length
Number of elements
Element 1*
Element 2
· · ·
Element n
4
2
Total length of area for each element
Real length
Element data format for repetition column
- Null element
Null flag 0x01
1
- Non-null value fixed-length element
Null flag 0x00
Data
1
Defined data length
- Non-null value variable-length element
Null flag 0x00
Real data length L
Data
1
2
L
Note: Length is in bytes.
* The start address of element data is not boundary-adjusted before being passed.

8.10.5 Example of a UOC

This section presents an example of table reorganization and the table unload operation using a UOC.

The example of UOC coding shown here is provided as a sample database. It is stored in the directory `$PDDIR/sample/sampleUOC`.

(1) Example of table reorganization using a UOC

During table reorganization, this example deletes data whose database registration date is year 2001 or earlier.

(a) Database table definition

```
  CREATE TABLE MEMBVER_DIRECTORY (MEMBER_NUMBER INTEGER,
                        MEMBER_NAME NCHAR(20),
                        MEMBER_ADDRESS NVARCHAR(100),
                        JOINED_DATE_AND_TIME DATE NOT NULL WITH
DEFAULT);
```

(b) Command format

```
pdrorg -k rorg -t MEMBVER_DIRECTORY control-information-file
```

(c) Contents of control information file

```
unload unload-data-file-name uoc_lib=library-name
unlduoc entry=old_data_delete
```

(d) Example of UOC coding

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <errno.h>
#include <pdutluoc.h>   ..........................................................1


void old_data_delete (
    UTL_UOC_INF         *uocinf             /* A(UOC interface area)      */   ...2
) {
    UTL_UOC_DATA_BUF    *addr_list;         /* A(data address list)       */
    unsigned char       *date_adr;          /* A(DATE)                    */
    static unsigned char cyear[2];          /* compare year               */

    switch(uocinf->req_cd){   ....................................................3
    case UTL_UOC_START:

/*● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ●*/
/*  START                                                          */
/*● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ●*/
        cyear[0] = 0x20;                    /* The 21st century           */
        cyear[1] = 0x01;                    /* (2001)                     */
        uocinf->rtn_code = UTL_UOC_NML;
        break;
    case UTL_UOC_EDIT:
```

```
/*● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ● ●*/
/*  EDIT                                                     */
/*● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ● ●*/
        addr_list = uocinf->data_adr;
        date_adr = addr_list->data[3];
        if (memcmp(date_adr,cyear,sizeof(cyear)) < 0){
            uocinf->unload_flg = UTL_UOC_ROWNOPUT;
        }else{
            uocinf->unload_flg = UTL_UOC_ROWPUT;
        }   ..............................................................4
        uocinf->rtn_code = UTL_UOC_NML;
        break;
    case UTL_UOC_END:
    case UTL_UOC_TERM:

/*● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ●*/
/*  END                                                    */
/*● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● ●*/
        uocinf->rtn_code = UTL_UOC_NML;
        break;
    default:
        strcpy(uocinf->err_msg,"Invalid request code");
        uocinf->rtn_code = UTL_UOC_ERR;
    }
    return;
}
```

Explanation:

1. Includes the header for UOC creation that is provided by HiRDB.
2. Receives the address of the UOC interface area as an argument.
3. Checks the call type and executes appropriate processing.
4. Checks the data (date) and sets the storage flag.

(2) Example of table unload operation using a UOC

This example edits `BLOB` data in CSV format when a table is unloaded. The example creates UOC data files based on one data item per file.

(a) Database table definition

```
CREATE TABLE IMAGE_TBL
          (COL1 INTEGER NOT NULL,
           COL2 CHAR(20) NOT NULL,
           COL3 INTEGER NOT NULL,
           COL4 CHAR(5) NOT NULL,
           COL5 BLOB(100K) IN (LOB_AREA) NOT NULL)
```

```
                    IN DATA_AREA;
```

(b) Command format

```
pdrorg -k unld -j -t IMAGE_TBL control-information-file
```

(c) Contents of control information file

```
unload (uoc) uoc_lib=library-name param='/usr/tmp/lob,/tmp/uoc_file'
unlduoc entry=blob_file_create
```

(d) Example of UOC coding

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <errno.h>
#include <fcntl.h>
#include <pdutluoc.h>


#define     INT         0xf0                /* INTEGER                    */
#define     CHAR        0xc4                /* CHAR                       */
#define     BLOB        0x92                /* BLOB                       */


void blob_file_create (
    UTL_UOC_INF         *uocinf             /* A(UOC interface area)      */
) {
    static  char         token[] = ",";     /* parameter token            */
    static  char         prefix[40];        /* path name prefix           */
    static  char         filename[80];      /* CSV file name              */
    static  char        *buff = NULL;       /* row edit buffer            */
    static  int          counter = 0;       /* row counter                */
    static  int          file_id = 0;       /* CSV file file ID           */
    int                  lobfid;            /* BLOB file                  */
    int                  leng;              /* total length               */
    int                  int_len;           /* INTEGER length             */
    int                  i;                 /* counter for column         */
    int                  rc;                /* return code                */
    int                 *nagasa;            /* BLOB size                  */
    void                *blobadr;           /* A(BLOB data)              */
    char                *wkadr;             /* work address              */
    UTL_UOC_DATA_BUF    *addr_list;         /* A(data address list)      */
    UTL_UOC_COLUMN_INFO *col_inf;           /* A(column information)     */
    char                *sepachr = ",";     /* separator character       */
    char                *linechr = "\n";    /* line field character      */
    char                 dmyname[16];       /* temporary name            */
    char                 blobfile[128];     /* BLOB file name            */

    switch(uocinf->req_cd){
    case UTL_UOC_START:

/*● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● 
● ● ● ● ●-*/
/*  START                                                            */
/*● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● 
● ● ● ● ●-*/
        if (uocinf->user_param == NULL){
            strcpy(uocinf->err_msg,"Required parameter not specified");
            goto OWARI;
        }


       wkadr = strtok(uocinf->user_param,token);
        if ((wkadr == NULL) ||
            (strlen(wkadr) >= sizeof(prefix))){
            strcpy(uocinf->err_msg,"Invalid parameter");
            goto OWARI;
        }
        strcpy(prefix,wkadr);   ..................................................1
        wkadr = strtok(NULL,token);
        if ((wkadr == NULL) ||
            (strlen(wkadr) >= sizeof(filename))){
            strcpy(uocinf->err_msg,"Invalid parameter");
            goto OWARI;
        }
        strcpy(filename,wkadr);   ................................................1


        file_id = create(filename,0666);
        if (file_id == -1){
            strcpy(uocinf->err_msg,strerror(errno));
            goto OWARI;
        }
        buff = malloc(1024);
        if (buff == NULL){
            strcpy(uocinf->err_msg,strerror(errno));
            goto OWARI;
        }


        counter = 0;
        strcpy(uocinf->err_msg,"FILE NAME:");
        strcat(uocinf->err_msg,filename);
        uocinf->rtn_code = UTL_UOC_DBG;
        break;
    case UTL_UOC_EDIT:
```

```
/*● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● */
/*  EDIT                                                    */
/*● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
● ● ● ● ● */
        counter++;
        col_inf = uocinf->dic_inf;
        addr_list = uocinf->data_adr;
        leng = 0;
        wkadr = buff;
        for(i = 0; i < uocinf->col_num; i++){
            switch(col_inf->colinf[i]->clm_type){

            case INT:   ........................................................2
                int_len = sprintf(wkadr,"%d",*((int*)addr_list->data[i]));
                leng += int_len;
                wkadr += int_len;
                break;
            case CHAR:   .......................................................2
                strncpy(wkadr,addr_list->data[i],col_inf->colinf[i]->clm_len);
                leng += col_inf->colinf[i]->clm_len;
                wkadr += col_inf->colinf[i]->clm_len;
                break;


            case BLOB:   .......................................................3
                sprintf(dmyname,"%010d",counter);
                strcpy(blobfile,prefix);
                strcat(blobfile,dmyname);
                lobfid = create(blobfile,0666);
                if (lobfid == -1){
                    strcpy(uocinf->err_msg,strerror(errno));
                    goto OWARI;
                }

                nagasa = (int *)((int)(addr_list->data[i]) + sizeof(int));
                blobadr = (void *)((int)(addr_list->data[i]) +
                                                (sizeof(int) * 2));
                rc = write(lobfid,blobadr,*nagasa);
                if (rc != *nagasa){
                    strcpy(uocinf->err_msg,strerror(errno));
                    goto OWARI;
                }

                rc = close(lobfid);
                if (rc == -1){
                    strcpy(uocinf->err_msg,strerror(errno));
                    goto OWARI;
                }
                strcpy(wkadr,blobfile);
                leng += strlen(blobfile);
                wkadr += strlen(blobfile);
                break;
            default:
                strcpy(uocinf->err_msg,"Not support data type");
                goto OWARI;
            }
            strcpy(wkadr,sepachr);
            leng += 1;
            wkadr++;
        }
        wkadr●;
        strcpy(wkadr,linechr);


        rc = write(file_id,buff,leng);
        if (rc != leng){
            strcpy(uocinf->err_msg,strerror(errno));
            goto OWARI;
        }
        uocinf->rtn_code = UTL_UOC_NML;
        break;

    case UTL_UOC_END:
    case UTL_UOC_TERM:

/*● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● 
● ● ● ● ●*/
/*  END                                                         */
/*● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● 
● ● ● ● ●*/
        if (buff != NULL){
            free(buff);
            buff = NULL;
        }

        if (file_id != 0){
            rc = close(file_id);
            file_id = 0;
            if (rc == -1){
                strcpy(uocinf->err_msg,strerror(errno));
                goto OWARI;
            }
        }
        uocinf->rtn_code = UTL_UOC_NML;
        break;

    default:
        strcpy(uocinf->err_msg,"Invalid request code");
        goto OWARI;
    }
    return;


OWARI:
    uocinf->rtn_code = UTL_UOC_ERR;
    return;
}
```

Explanation:

1. Expands the user parameters specified in the control information file into variables within the UOC.
2. Outputs the data for an `INTEGER` or `CHAR` column in characters to a UOC data file.
3. Saves the data for a BLOB column in rows to a file and then outputs the file name to a UOC data file.

8.11 Notes on executing pdrorg on special tables

8.11.1 Table with an abstract data type

(1) Unloading a table

When unloading a table that contains an abstract data type, note the following:

(a) Unloading only the table for which an abstract data type with a LOB attribute is defined

- You can unload the table only if the plug-in defined for the abstract data type column has an unload facility or a constructor parameter reverse creation function.
- To execute the unload operation only, specify the `-j` option.

(b) Unloading a table to migrate it to another table or system

- To transfer data from one table to another table in the same system, the source table is unloaded. With some plug-ins, you can execute such a data transfer only if the plug-in has the unload facility or constructor parameter reverse creation function. To determine whether or not table migration is permitted, see the applicable plug-in documentation.
- Transferring data from one table to another table in the same system may be supported only when the plug-in has the constructor parameter reverse creation function. For details about whether or not data can be transferred, see the applicable plug-in documentation.
- To unload a table when the plug-in has a constructor parameter reverse creation function, you need to specify the constructor parameter reverse creation function in the `unld_func` statement.

(c) Changing the partitioning conditions of a table with an abstract data type

- To modify a table's partitioning conditions, first unload the table. With some plug-ins, you can change the table partitioning conditions only if the plug-in has an unload facility or constructor parameter reverse creation function. To determine whether or not you can change table partitioning conditions, see the applicable plug-in manual.
- To unload a table when the plug-in has a constructor parameter reverse creation function, you need to specify the constructor parameter reverse creation function in the `unld_func` statement.

(2) Reloading to a table

When reloading data to a table that contains an abstract data type, note the following:

(a) Table for which an abstract data type with a LOB attribute is defined

If you have specified the `-j` option during the unload operation, be sure to specify the `-j` option also during the reload operation.

(b) Another table or a table in another system

- To reload data to another table or to a table in another system, specify the `tblname` statement.
- To use the unload data file obtained by specifying a constructor parameter reverse creation function during an unload operation, you need to specify the constructor function in the `reld_func` statement. This constructor parameter must correspond to the constructor parameter reverse creation function that was specified during the unload operation. For the correspondence between constructor function and constructor parameter reverse creation function, see the applicable plug-in manual.

(c) Changing the partitioning conditions of a table with an abstract data type

If you have unloaded a table by specifying a constructor parameter reverse creation function during the unload operation, you need to specify the constructor function in the `reld_func` statement during the reload operation after changing the partitioning conditions. When specifying a constructor function to reload a table, you can use only the unload data file that was obtained by specifying the constructor parameter reverse creation function.

The specified constructor function must correspond to the constructor parameter reverse creation function that was specified during the unload operation. For details about the correspondence between constructor function and constructor parameter reverse creation function, see the applicable plug-in manual.

(3) Relationship between pdrorg functions and the control information file

Table 8-34 shows the relationship between `pdrorg` functions and the control information file.

Table 8-34: Relationship between pdrorg functions and the control information file

pdrorg function	**-k option**	**-j option**	**Control information file**				
			unload stmt	**lobunld stmt**	**tblname stmt**	**unld_func stmt**	**reld_func stmt**
Reorganization of LOB column structure base table only (1)	`rorg`	—	R	N	N	N	N
	`unld`						
	`reld`						

pdrorg function	-k option	-j option	Control information file				
			unload stmt	lobunld stmt	tblname stmt	unld_func stmt	reld_func stmt
Reorganization of LOB columns only (3)	`rorg`	—	N	R	N	N	N
	`reld`						
Reorganization of LOB column structure base table, LOB column storage RDAREAs, and LOB attribute storage RDAREAs (1, 2, and 3)	`rorg`	R	R	N	N	O[2]	O[2]
	`unld`					O[3]	N
	`reld`					N	O[3]
Modification of table partitioning conditions1 (1, 2, and 3)[1]	`unld`	R	R	N	N	O	N
	`reld`					N	O
Migration to another table or system1 (1, 2, and 3)[1]	`unld`	R	R	N	N	O	N
	`reld`				R	N	O

R: Required.

O: Optional.

N: Cannot be specified.

—: Not applicable.

stmt: statement

Note

A number in parentheses in the `pdrorg` function column indicates the RDAREA being processed, which corresponds to the following table definition:

```
CREATE TABLE ...
    C1 INT,
    C2 SGMLTEXT ALLOCATE(sgmltext IN lob01)      2
    C3 BLOB(10K) IN LOB02                        3
    IN USER01                                    1
```

[1] With a HiRDB/Parallel Server, if the table is divided and stored at multiple back-end servers, the `-g` option must be specified.

[2] The `unld_fund` and `reld_func` statements must both be specified.

[3] If specified for the unload operation, also specify it for the reload operation.

(4) Reloading when abstract data type definitions differ between the unloaded table and the table to be reloaded

Abstract data type definitions may not match between the unloaded table and the table to be reloaded due to data migration to another table or plug-in upgrading. In this case, when executing a reload operation, note the following:

1. You can reload to the table if both of the following conditions are met:
   - The unload data file to be used was unloaded by specifying a constructor parameter reverse creation function in the `unld_func` statement during the unload operation.
   - The `reld_func` statement specifies the constructor function with a parameter count and attribute that match those of the constructor parameter reverse creation function specified during unload operation.
2. If no constructor function has been specified, you can execute a reload operation if the following information is the same for both the unload and reload tables:
   - Name of abstract data type
   - Plug-in ID
   - Number of attributes
   - Attribute's data type and definition length
   - LOB attribute storage RDAREA

(5) Reloading when table partitioning conditions have changed

With some plug-ins, you need to specify a constructor parameter reverse creation function during the unload operation to change the table partitioning conditions before reloading the table. In this case, unload the table specifying the constructor parameter reverse creation function in the `unld_func` statement, change the table partitioning conditions, and then reload the table, specifying the constructor function in the `reld_func` statement.

In this case, note the following:

- With a HiRDB/Parallel Server, if the table is partitioned and stored at multiple back-end servers, specify the `-g` option during both the unload and reload operations.

- When reloading a table in units of RDAREAs, if the table partitioning conditions do not match, an error results.
- For a table with LOB columns or columns of abstract data type including the LOB attribute, specify the `-j` option.

(6) When data was unloaded without using the constructor parameter reverse creation function

When data is unloaded without using the constructor parameter reverse creation function, do not register or delete plug-ins until reloading is completed.

8.11.2 Falsification prevented table

(1) Limitations

Even when a deletion prevention duration is specified, `pdrorg` rearranges data; therefore, row data is deleted once. Falsification of data is prevented by executing the unload and reload operations one after another. You can perform only table reorganization (`-k rorg`) on a falsification prevented table. The table unload (`-k unld`) and reload (`-k reld`) operations are permitted only under specific conditions.

(a) For table reorganization (-k rorg)

During table reorganization, the following functions are disabled:

- Reorganization using a UOC
- Reorganization with synchronization point specification

(b) For table unload operation (-k unld)

Basically, the table unload operation is prohibited for falsification prevented tables. However, you can unload a table for which the `-W` option has been specified.

(c) For table reload operation (-k reld)

Basically, the table reload operation is prohibited for falsification prevented tables. However, if a reload operation for table reorganization (`-k rorg`) terminates abnormally, you can re-execute that reload operation (`-k reld`). During such a re-execution, do not change any control statement or option other than the `-k` option (if execution with the `unload` and `lobunld` statements both specified results in an error, re-execute the operation with only one of those statements specified).

(2) Execution conditions

Before executing table reorganization (`-k rorg`) or a table reload operation (`-k reld`), you must use the `pdhold` command to shut down the RDAREA that contains a falsification prevented table (to protect its data from being compromised).

(3) Reload-not-completed data status

During reorganization of a falsification prevented table, the reload operation may not be completed due to an error. This status is called the *reload-not-completed data status*.

You can determine whether or not an RDAREA is in reload-not-completed data status from the results of RDAREA logical analysis or table analysis by `pddbst`. When the RDAREA is in reload-not-completed data status, you cannot use any of the following `pdrorg` functions:

- Table reorganization (`-k rorg`)
- Batch index creation (`-k ixmk`)
- Index re-creation (`-k ixrc`)
- Index reorganization (`-k ixor`)

8.12 Rules and notes

(1) Rules

(a) Executing the utility

- You can execute the database reorganization utility only while HiRDB is active.
- Execute the database reorganization utility at the server machine containing the single server or the server machine where the system manager is located.
- To execute `pdrorg`, you set the `LANG` environment variable. To use in a `pdrorg` execution environment character codes that are not supported by the OS, you must set the `PDLANG` environment variable. For details about `LANG` and `PDLANG`, see the *HiRDB Version 8 UAP Development Guide*.

(b) Whether the utility can be executed

Whether or not the database reorganization utility can be executed depends on the open attribute of the table, index, and RDAREA containing LOB columns and the status of the RDAREA; for details, see Appendix *C. RDAREA Status During Command Execution*.

To reorganize a data dictionary table, place the RDAREA containing the data dictionary table in shutdown status by the `pdhold` command.

(c) Maximum number of concurrently executable utilities

The maximum number of database reorganization utilities that can be executed at any one time depends on the value of the `pd_utl_exec_mode` operand in the system common definitions.

`pd_utl_exec_mode=0`

A maximum of 32 utilities can be executed concurrently.

`pd_utl_exec_mode=1`

The value of the `pd_max_users` operand determines the maximum number of concurrently executable utilities.

Note that during execution of `pdreclaim` and `pdpgbfon`, the maximum number of `pdrorg` commands that can be executed concurrently is reduced because `pdreclaim` and `pdpgbfon` call `pdrorg` internally. For example, if `pd_utl_exec_mode=0` is specified, theoretically a maximum of 32 `pdrorg` commands can be executed; however, if 10 `pdreclaim` commands are executing, the maximum number of `pdrorg` commands that can be executed concurrently is reduced to 22.

(d) Tables and indexes during utility processing

- A table or index cannot be accessed by another UAP or utility while it is being processed by pdrorg because the target resource is locked. In the case of -k unld, whether or not a table or index is accessible depends on the unldenq operand specification in the option statement. For details, see *8.9.16 option statement (specification of data processing information)*.

- If you shut down the RDAREAs containing the table to be reorganized, a UAP attempting to access that table results in a shutdown error. This prevents the utility's execution from being placed in lock-release wait status due to UAP execution, or prevents a UAP from being placed in lock-release wait status due to utility execution. For details about pdrorg's lock mode, see *B.2 Lock mode for utilities*.
- Do not execute a definition SQL statement on a table or index while it is being reorganized. If a definition SQL statement is executed, pdrorg terminates abnormally. When reorganization is performed in units of schemas, all tables and indexes that belong to the corresponding schema become the target of processing. To prevent execution of a definition SQL statement on a table or index being reorganized, you must place the RDAREAs containing the table and indexes being reorganized on shutdown status with the pdhold command. To reference another table in the same RDAREA during table reorganization, place it in reference-possible shutdown status by the pdhold command.
- If the database load utility is executed on a table or index in an RDAREA that is being processed by a definition SQL statement, the table or index will be placed in lock-release wait.
- While a table is being reorganized, do not change its definitions or execute a UAP or another utility. If a UAP or another utility is executed, it will result in a lock error. If unload and reload operations are executed separately with the shell, the number of table data items may be set to zero, or the updating results may become invalid.

(e) Changing partitioning key ranges

If the partitioning key ranges are changed before a reload operation and the unloaded data does not fit in a redefined range, an error results and processing will terminate when that data is encountered during the reload operation. In this case, redefine the partitioning key ranges so that all the unloaded data fits within defined ranges, and reload the data.

(f) Log-acquisition mode during utility execution on an extracted database subject to data linkage

To execute the database reorganization utility on an extracted database subject to data linkage, specify n or p in the -l option (to execute in no-log mode or pre-update log acquisition mode). If you execute the utility in the log acquisition mode, conformity may be lost between the extracted database and the target database because only a portion of the data updated in the extracted database is sent to the target database.

(g) Facility for conversion to a DECIMAL signed normalized number

The database reorganization utility does not normalize the sign part of `DECIMAL` type, whether or not the facility for conversion to a DECIMAL signed normalized number is used. Therefore, if the unload data file was created by a HiRDB system that does not use the facility for conversion to a DECIMAL signed normalized number, you cannot reload this file to a table in a HiRDB system that uses this facility.

To normalize the sign part of a `DECIMAL` type in a database, create an unload data file with the `-W` option specified, then store the data again using the database load utility. For details about the facility for conversion to a DECIMAL signed normalized number, see the *HiRDB Version 8 System Operation Guide*.

(h) Output destination of index information files

The output destination of index information files depends on the type of `pdrorg` processing and the specified control statements, as shown in the following table:

Type of pdrorg processing	**Control statement**			**Output destination of index information files**
	index statement	**idxname statement**	**idxwork statement**	
`-k rorg`, `-k reld`, `-k ixmk`	Specified*	N	—	Index information file specified in the `index` statement
	Not specified	N	Specified	Directory specified in the `idxwork` statement
	Not specified	N	Not specified	`/tmp`
`-k ixrc`, `-k ixor`	N	Specified	Specified	Directory specified in the `idxwork` statement
	N	Specified	Not specified	`/tmp`
	Specified	N	—	Index information file specified in the `index` statement

N: Cannot be specified.

—: Specification has no effect on the output destination of index information files.

* For a row-partitioned table, only the specified RDAREAs are being processed; therefore, the items with `index` statement not specified apply to the unspecified RDAREAs.

(i) Reorganizing a rebalancing table

If you have added an RDAREA to a rebalancing table but have not executed `pdrbal` on the rebalancing table (return code = `0`), you cannot reorganize the added RDAREA (in units of RDAREAs).

Table 8-35 shows whether or not `pdrorg` can be executed on an RDAREA that has been added to a rebalancing table.

Table 8-35: Whether or not pdrorg can be executed on an RDAREA that has been added to a rebalancing table

-k option		Table status				
		RDAREA addition up to pdrbal execution		during pdrbal processing[*] (return code = 4)		After completion of pdrbal processing (return code = 0)
		Existing	Added	Existing	Added	
`unld`		Y	—	Y	Y	Y
`rorg,` `reld`	By table	Y	—	Y[*]	Y[*]	Y
	By RDAREA	Y	—	—	—	Y
`ixmk`		Y	—	Y	Y	Y
`ixrc`		Y	—	Y	Y	Y
`ixor`		Y	—	Y	Y	Y

Existing: Existing RDAREA specified in the table definition (`CREATE TABLE`)

Added: RDAREA added by `ALTER TABLE ADD RDAREA`

Y: `pdrorg` can be executed.

— : `pdrorg` cannot be executed.

[*] To reload data while rebalancing the table, specify the `-g` option. If the table contains a LOB column or a column of an abstract data type with the LOB attribute, also specify the `-j` option. When a table is reorganized with the `-g` option specified, data is rearranged but its rebalancing is not completed. Therefore, make sure that `pdrabl` is executed after the reload operation and that its return code is `0`.

To add an RDAREA to an unloaded table and then reload it, use the following procedure:

1. Execute `pdrorg` (`-k unld`).
2. Add an RDAREA with `ALTER TABLE ADD RDAREA`.
3. Delete all rows with `PURGE TABLE`.
4. Execute `pdrbal`.
5. Execute `pdrorg` (`-k reld`)

When executing the previous procedure, be sure to specify the `-g` option in `pdrorg`. Otherwise, an error results. For a rebalancing table with LOB columns or columns of abstract data type, also specify the `-j` option.

(j) Processing an RDAREA in frozen update status

Executing `-k rorg` (reorganization) or `reld` (reload operation) on an RDAREA that is in frozen update status results in an error, because the utility cannot update the management area for the user LOB RDAREA. An attempt to reorganize or reload only a LOB column structure base table also results in the same error.

(k) Processing an external table

`pdrorg` executed on an external table terminates with an error.

(l) Reloading a table with SEGMENT REUSE specified

When reloading a table, the `SEGMENT REUSE` specification is ignored.

(m) About a table that is row-partitioned by the hash function with the rebalancing facility

If a table that has been FIX hash-partitioned by the rebalancing facility is to be reorganized, and if the partitioning conditions are changed between the unload and the reload processes, a shortage of RDAREA may occur due to lack of available segments even though there are free pages in the table storage RDAREA. This is because a change in the storage conditions requires data to be reloaded into a different segment from the one used during unloading, but there is no new segment that can be allocated. In such a case, expand the RDAREA before you execute reloading.

(n) Compatibility in unload data files between the 32-bit and the 64-bit version of HiRDB

Data can be migrated between the 32-bit version and the 64-bit version of HiRDB by reloading the unload data files created by the 32-bit or 64-bit HiRDB. The migration conditions are the same as for migration between HiRDBs that use the same bit mode.

(2) Notes

(a) Return code

The following are the `pdrorg` utility's return codes:

`0`: Normal termination

`4`: Normal termination (some of the processing was skipped)

`8`: Abnormal termination

(b) Collecting a synchronization point dump

The system does not collect synchronization point dumps during the execution of `pdrorg`, because a transaction is not settled until one process is completed. One process means an unload operation, a reload operation, or index creation.

If other UAPs are executed at the same time as `pdrorg` and a system failure occurs during processing, the time required for restart increases. For this reason, you should not execute UAPs while `pdrorg` is executing, if possible.

For a reorganization or reload operation with the synchronization point specification, you can collect synchronization point dumps at intervals of specified number of lines. This reduces the time required for restart in the event of abnormal termination, compared with the reorganization or reload operation with no synchronization point dump collected.

(c) Canceling processing during utility execution

To cancel processing during execution of `pdrorg`, use the `pdcancel` command. To treat the `pdrorg` command as resulting in a no-response error (such as when a routine reorganization job should finish within a known amount of time, but it does not) and forcibly terminate the command, redirect the display results of the `pdls` command (with `-d rpc -a` specified) to a file and then execute the `pdcancel -d` command.

In this case, processing will be rolled back. For details about the database status and recovery method, see *8.13 Database status in the event of an error and recovery method*.

If you are using the facility for predicting reorganization time and you forcibly terminate `pdrorg` by a signal interrupt such as the `kill` command, the database management table cannot be updated. To forcibly terminate `pdrorg` while the facility for predicting reorganization time is used, make sure that you use the `pdcancel` command.

(d) Modifying definition information during utility execution

Once a table has been unloaded, do not specify the following definition SQL statements until the table has been reloaded:

- `ALTER TABLE`
- Re-creation of table definition with `DROP TABLE` and `CREATE TABLE`

However, you can execute these definition SQL statements when changing the partitioning key partitioning condition or the hash function.

(e) Results of utility execution

You can check the result of the database reorganization utility using the database condition analysis utility.

(f) Changing partitioning conditions for a partitioning key during reloading

An error will occur during a reload operation if you specify either of the following options when changing the partitioning key partitioning condition:

- `rorg` in the `-k` option
- `-r` option

(g) Using a multi-volume MT

Because MTguide is used for the volume switching operation when multi-volume magnetic tape is used, MTguide must be installed at the server machine.

(h) Status of an index during the reinitialization of the RDAREA

If you have reinitialized an RDAREA by the database structure modification utility, any index related to the RDAREA may be placed in incomplete status. If a non-partitioning key index is placed in incomplete status due to reinitialization, reorganize the table in the batch index creation mode in units of tables. If you create an index in batch mode using the index file that was output before reinitializing the RDAREA, the database may be damaged.

(i) Using the system switchover facility

If you are using the system switchover facility, we recommend that you execute `-k unld` and `-k reld` instead of executing `-k rorg` for table reorganization. This method enables you to continue the reload operation at the target system using the unload data file on which the unload operation has been completed. In such a case, use for the unload data file a magnetic tape device (such as a DAT) or a HiRDB file created by a character string special file. If you use a regular file for block input/output operations to create an unload data file, the contents of the unload data file cannot be guaranteed even when the unload operation terminates with return code `0`, because the OS buffer is deleted during system switchover.

(j) Abnormal termination of pdrorg during creation of an unload file

If `pdrorg` terminates abnormally while an unload data file is being created, invalid files may remain. Re-executing `pdrorg` when this happens results in the remaining files being overwritten. If you do not re-execute the utility, you should delete these files. If the files to be deleted are regular files, use the OS's `rm` command to delete them; if they are HiRDB files, use the `pdfrm` command to delete them.

If `pdrorg` terminates abnormally while a table is being reorganized or reloaded, the processing is rolled back and data is deleted from the database. To restore the database, use the unload data file created by unload processing to execute reloading only. If multiple tables are output one after another to a single unload data file, the unload data file is overwritten when unloading of another table is performed after abnormal termination of `pdrorg`. As a result, data is lost from both the unload data file and the database. Therefore, if the database cannot be restored from its backup, make sure that you specify a separate unload data file for each table.

(k) Unloading and reloading between systems

The following notes apply to unloading and reloading tables between systems:

- You cannot execute an unload or reload operation between the systems with a different magnitude of byte size.
- You cannot execute an unload or reload operation between the systems with different character organizations.
- An unload data file contains HiRDB version-specific control information in addition to the row data. Upward compatibility is supported; that is, an unload data file created by one version of HiRDB can be reloaded to a later version of HiRDB, but downward compatibility is not guaranteed. To reload data to a later version of HiRDB, use the unload data file created with the `-W` option specified.
- If a column of an abstract data type has been defined in the table, you must use the constructor parameter reverse creation function to unload the table.

(l) Modifying the partitioning key range or the hash function

When the partitioning key ranges are changed or the hash function is changed, the utility divides data and stores it into RDAREAs based on the changed partitioning conditions only at the following times (excluding the cases in which the partitioning key column is changed or the `-j` option is omitted for a table containing a LOB column):

- When reloading with a HiRDB/Single Server
- When reloading from an unload data file with the `-g` option specified with a HiRDB/Parallel Server

For details about how to switch among key range partitioning, flexible hash partitioning, and FIX hash partitioning, see the *HiRDB Version 8 System Operation Guide*.

(m) Unloading and reloading using identical table definitions

If the table definitions are the same, a table unloaded from a HiRDB/Single Server can be reloaded to a table in a HiRDB/Parallel Server, and vice versa. If the table contains a `LOB` column, you need to specify the `-j` option during the unload and reload operations.

If a table to be unloaded from a HiRDB/Parallel Server is partitioned and stored in multiple servers, you need to specify the `-g` option to create a single unload data file before you can reload it to a table in a HiRDB/Single Server.

If the table definitions are different or if you want to use the unload data files that are output for individual servers, execute the unload operation specifying the `-W` option, then use the database load utility to load the data. In this case, if `-W bin` is specified, but the `-j` option is omitted, the null value is set in any LOB columns. When you transfer a `LOB` column, specify both options, `-W bin` and `-j`. If a cluster key index has been defined for a row-partitioned table, data in the unload data file is sorted by the cluster key in each RDAREA, but it is not sorted by the cluster key in the entire table.

In such a case, a clustering order error occurs. Therefore, you must specify the `-x` option in `pdload` or perform processing in units of RDAREAs.

(n) Using large files

Use of large files enables you to handle files whose size is 2 gigabytes or greater. Table 8-36 describes whether or not `pdrorg` supports large files. The maximum file size that can be created by the process is determined by the settings of the operating system in use. For the limit value of system resources for the HiRDB administrator and root user, either specify a value that is greater than the size of file that is created or do not limit the value. Special attention is needed under AIX-5L because its default file size is 1 gigabyte. You can check the limit value of system resources with the OS's `limit` or `ulimit` command. To change the limit value for file size in AIX 5L, you must also change the `/etc/security/limits` file. For details, see the applicable OS and shell documentation. Because HiRDB is an INIT startup process, you must restart the operating system before a value changed by the root user can take effect.

Table 8-36: Whether or not large files can be used for pdrorg processing

File type	Availability of large file system
Unload data file	Y
LOB data unload file	Y
Index information file	Y
Work file for sorting	Y
MT attribute definition file	N
Process results file	Y

Y: Can be used.

N: Cannot be used.

(o) Reloading a table with columns of the abstract data type

- If you have re-initialized an RDAREA that stores a LOB column structure base table, LOB columns, or LOB attribute, you can reload only the unload data file created with the `-j` option specified. In such a case, you must also specify the `-j` option during reloading.
- To process a table in units of tables without specifying the `-g` option, you need as many dictionary server processes as there are back-end servers containing the table.

(p) Whether batch index creation and index re-creation are supported for a plug-in index

The following table shows whether batch index creation and index re-creation are supported for a plug-in index:

Provided function	Batch index creation	Index re-creation
Plug-in index delayed batch creation facility	Executable	Executable
Batch plug-in index creation partial recovery facility	Executable	Executable
None of the these facilities available	Not executable	Executable

To execute batch index creation on a plug-in index (by specifying `-k ixmk`), the plug-in must provide the batch plug-in index creation partial recovery facility. If the plug-in does not provide this facility, you need to re-create the index (by specifying `-k ixrc`).

If batch index creation that uses the index information file created by the plug-in index delayed batch creation facility terminates normally, the system deletes the that index information file. Note that the index information file output directory (directory with `pd_plugin_ixmk_dir` specified in the server definition) used by this facility should not be used as a file I/O area for the database reorganization utility unless the utility processing is batch index creation (`-k ixmk`).

(q) Index updated by the utility using the differential index function of the HiRDB Text Search Plug-in

When using the differential index function of the HiRDB Text Search Plug-in, the database reorganization utility updates the following indexes:

Index creation method	Specification of environmental variable PDPLUGINNSUB		
	Y	N	Not specified
Re-creating an index (`-k ixrc` specified)	M	M	M
Creating an entire index (`-k ixmk` specified)	M	M	M
Creating an index for additional data (`-k ixmk` specified)	S	M	S

M: Updates the `MASTER` index.

S: Updates the differential index.

(r) List created using a table being reorganized or reloaded

If you have created a list using a table being reorganized or reloaded, searching this list after the reorganization or reload operation results in the following events:

- A wrong row is retrieved.
- Specified row is not found.

In this case, you need to re-create the list before searching the list.

(s) Locked resources required per server during the execution of the database reorganization utility

During the execution of the database reorganization utility, each server requires the following amount of locked resources:

- Resources used for lock control outside the transaction
  $X = 2 \times (a + b + c + 1) \times (b \times d)$

  Note

  The value of the `pd_lck_until_disconnect_cnt` operand in the system definitions must be at least the value of *X*.

- Resources used for lock control within the transaction
  $Y = e + f + g$

  Note

  The value of the `pd_lck_pool_size` operand in the system definitions must be at least $\uparrow Y \div X \uparrow$ KB. Because the utility's preprocessing requires (209 + *A*) resources, at least (209 + *A*) resources are required for *Y*. The resources allocated during preprocessing are released before reorganization begins; therefore, if the obtained value of *Y* is (209 + *A*) or greater, use that value.

Explanation:

a: Number of table storage RDAREAs

b: Number of index storage RDAREAs

c: Number of LOB column storage RDAREAs

d: Number of indexes

e: Number of segments used by RDAREAs for table

f: Number of segments used by RDAREAs for index

x: 4 for a HiRDB in the 32-bit mode, and 6 for a HiRDB in the 64-bit mode

(t) Use of DVD-RAM devices

You can use a DVD-RAM device for unload data files for the `pdrorg` command. Note that the supported device depends on the operating system in use.

To use as regular files, you can create a file system on a medium in the same manner as conventional devices such as magnetic disks.

To use as HiRDB files, you must specify a sector length during creation of the HiRDB file system area (during the `pdfmkfs` command's execution).

(u) File media available during utility execution

The table below shows the file media that are available during execution of `pdrorg`. When regular files are used, file open processing accompanies the use of the `maxfiles`, `nfile`, and `nflocks` operating system parameters (kernel parameters).

File	Regular file	HiRDB file	Blocked fixed-length tape	Blocked variable-length tape
Unload data file	Y	Y	Y	Y
Unload data file for LOB data	Y	Y	Y	Y
Index information file	Y	If	N	N
Work file for sorting	Y	N	N	N
Process results file	Y	N	N	N

Legend:

Y: Can be used.

If: Can be used if the delayed batch index creation facility is used.

N: Cannot be used.

(v) Messages displayed in the command execution window

`pdrorg` outputs progress messages to the standard output during processing. In the event of an error, `pdrorg` outputs an error message to the standard error output. If `pdrorg` is executed in an environment in which output to the standard output and standard error output is suppressed, `pdrorg` may be placed in the message output wait status and stop responding, or may output the `KFPL20003-E` message to the message log file and then terminate abnormally. Therefore, you should not execute `pdrorg` in an environment in which messages cannot be output to the standard output or standard error output. Note that the sequence and number of messages that are output to the standard output and standard error output may not match those in the message log file and `syslogfile`. To obtain the correct messages, view the message log file or `syslogfile`.

(w) Reorganizing a shared table

When a shared table is reorganized, the system places the RDAREAs containing the shared table and shared indexes defined for the target table in the `EX lock` mode. If the corresponding RDAREAs contain other tables and indexes, these tables and indexes cannot be referenced or updated. For details about the lock mode when a shared table is reorganized, see Appendix *B.2 Lock mode for utilities*.

(x) Reorganizing a table for which referential constraints or check constraints have been defined

When reorganization or reloading is to be executed, for details about whether or not the check pending status can be set for a table for which referential constraints or check constraints have been defined, see *8.9.15 constraint statement (specification of check pending status)*. For details about the check pending status, see the manual *HiRDB Version 8 Installation and Design Guide*.

When all of the following conditions are satisfied, a table in check pending status cannot be reorganized (`-k rorg`):

- Reorganization is by table (`-r` option is omitted)
- The table has been partitioned by flexible hash partitioning or by a matrix partitioning including flexible hash partitioning.
- Either a HiRDB/Single Server, or a HiRDB/Parallel Server with the `-g` option specified is used.

Whether or not the table is in check pending status is determined from the value of the `CHECK_PEND` or `CHECK_PEND2` column in the data dictionary table (`SQL_TABLES` table) (if the column value is `'C'`, the table is in check pending status).

(y) Relationship with the facility for predicting reorganization time

The history of table and index reorganization is applied to the result of reorganization time prediction.

If `pdrorg` has terminated abnormally, executing `pddbst`'s condition analysis result accumulation facility results in an invalid prediction result* because the reorganization timing cannot be predicted in the reorganization completed status. Therefore, if `pdrorg` terminates abnormally, re-execute `pdrorg` to terminate it normally, then execute `pddbst`'s condition analysis result accumulation facility.

* For example, if rollback occurs due to abnormal termination during reloading, empty database information may be accumulated and used for prediction.

(3) Using a file that contains a BOM

If you selected `utf-8` as the character encoding in the `pdsetup` command, you may be able to use a file with a BOM as the input file for `pdrorg`. Table 8-37 shows the files with a BOM in `pdrorg`. Note that even when a file with a BOM is used as the input file to `pdrorg`, the BOM is skipped. No BOM is included in the file that is output by `pdrorg`.

Table 8-37: Whether or not files with a BOM can be used in pdrorg (applicable to UTF-8)

Control statement	Input file	Use of file with a BOM
●	Control information file	Y
`emtdef`	MT attribute definition file	N
`unload`	Unload data file	N
	EasyMT information file	N
`index`	Index information file	N
`lobunld`	LOB data unload file	N
	EasyMT information file	N

Legend:

Y: Can be used

N: Cannot be used

● : Not applicable

8.13 Database status in the event of an error and recovery method

If an error occurs during execution of `pdrorg`, check Tables 8-38 and 8-39 to determine the progress of the processing and take appropriate action. In the case of reorganization (`-k rorg`) that involves consecutive execution of unload and reload operations, if an error occurs during a reload operation, the data for the corresponding table is placed in reload-not-completed data status, in which case only the reload operation (`-k reld`) can be re-executed. You cannot use `pdload` on such a table that is in reload-not-completed data status to execute data loading in the addition mode, nor can you use `pdrbal` to execute rebalancing. If reloading is not possible because the unload data file has been deleted, you can release the reload-not-completed data status by executing the `PURGE TABLE` statement or `pdload` in the creation mode.

Table 8-38: Message output during execution of pdrorg and action to be taken (1/2)

Option value during execution of pdrorg	**Message output during execution of pdrorg**							
	712 and 732 uld	**714 and 734 uld**	**721**	**712 and 732 rld**	**714 and 734 rld**	**725**	**715**	**716**
`-k rorg -j` omitted	8.13.1	8.13.1	8.13.2	8.13.2	8.13.2	—	8.13.2	8.13.2
-k `rorg -j` specified								
`-k unld -j` omitted			—	—	—		—	—
`-k unld -j` specified								
`-k reld -j` omitted	—	—	8.13.2	8.13.2	8.13.2		8.13.2	8.13.2
-k `reld -j` specified								
`-k ixmk`			—	—	—		8.13.3	8.13.3
-k `ixrc`						8.13.3		
`-k ixor`						8.13.4	8.13.4	8.13.4

Legend:

—: Not applicable

712 and 732 uld: Data unloading started

712 and 732 rld: Data reloading started

714 and 734 uld: Data unloading completed

714 and 734 rld: Data reloading completed

715: Index creation started

716: Index creation completed

721: Data deleted

725: Index information searched

Note

Messages are abbreviated. For example, `712` means `KFPL00712-I` and `732` means `KFPL00732-I`.

Table 8-39: Message output during execution of pdrorg and action to be taken (2/2)

Option value during execution of pdrorg	**Message output during execution of pdrorg**				
	712 and 732 uld	**714 and 734 uld**	**721**	**712 and 732 rld**	**714 and 734 rld**
`-k rorg -j` omitted	8.13.5	8.13.5	8.13.6	8.13.6	8.13.6
`-k rorg -j` specified	—	—	—	—	—
`-k unld -j` omitted	8.13.5	8.13.5			
`-k unld -j` specified	—	—			
`-k reld -j` omitted	8.13.5	8.13.5	8.13.6	8.13.6	8.13.6
`-k reld -j` specified	—	—	—	—	—
`-k ixmk`					
`-k ixrc`					
`-k ixor`					

Legend:

— : Not applicable

712 and 732 uld: Unloading of LOB data started

712 and 732 rld: Reloading of LOB data started

714 and 734 uld: Unloading of LOB data completed

714 and 734 rld: Reloading of LOB data completed

721: LOB data deleted

Note

Messages are abbreviated. For example, `712` means `KFPL00712-I` and `732` means `KFPL00732-I`.

8.13.1 Error when unloading

During an unload operation, an error does not affect the contents of the database. If an error occurs, you can simply re-execute the unload operation after eliminating the cause of the error.

8.13.2 Error when reloading

Table 8-40 shows the database status and recovery method in the event of an error during a reload operation. The recovery method depends on the options specified for execution of the database reorganization utility. If an error occurs during reload operation, check the specified options and the message output immediately before the error occurred, as shown in Table 8-40, and then recover the error.

Table 8-40: Database status and recovery method in the event of an error during reload operation

Option		Status/ Recovery method	Message output immediately before error occurred					
-l	-i		None	721	712 and 732	714 and 734	715	716
`a`	`s`	Status	Status before exe	Null*2	Null*2, *4	Table crtn cmpled	—	—
		Recovery	Re-exe	Re-exe reload*1	Re-exe reload*1	—		
	`c`	Status	Status before exe	Null*2	Null*2, *4	Table crtn cmpled	Index not created	Index crtn cmpled
		Recovery	Re-exe	Re-exe reload*1	Re-exe reload*1	—	Create index	Unneces sary

Option		Status/ Recovery method	Message output immediately before error occurred					
-l	-i		None	721	712 and 732	714 and 734	715	716
p	s	Status	Status before exe	Null*2	Null*2, *4	Table crtn cmpled	—	—
		Recovery	Re-exe	Re-exe reload*1	Re-exe reload*1	—		
	c	Status	Status before exe	Null*2	Null*2, *4	Table crtn cmpled	Index not created	Index crtn cmpled
		Recovery	Re-exe	Re-exe reload*1	Re-exe reload*1	Create index	Create index	—
n	s	Status	Not guaranteed	Null*2	Not guaranteed	Table crtn cmpled	—	—
		Recovery	Re-exe after recover database	Re-exe reload after recover database	Re-exe reload after recover database	—		
	c	Status	Not guaranteed	Null*2	Not guaranteed	Table crtn cmpled	Index not created	Index crtn cmpled
		Recovery	Re-exe after recover database	Re-exe reload after recover database	Re-exe reload after recover database	Re-exe reload after recover database*3	Re-exe reload after recover database*3	—

Legend:

721: Row data deleted

712 and 732: Data reloading started

714 and 734: Data reloading completed

715: Index creation started

716: Index creation completed

—: Not applicable

cmpled: completed

crtn: creation

exe: execution

Re-exe: Re-execution

recover: recovering

Note

Messages are abbreviated. For example, `712` means `KFPL00712-I` and `732` means `KFPL00732-I`.

*1 If an error occurs during execution in the log acquisition mode or the pre-update log acquisition mode, the table data may become empty. If you execute an unload operation, null data is unloaded and the table data is lost. In such a case, re-execute the reload operation only.

If reorganization with the `unload` and `lobunld` statements specified and without the `-j` option results in re-execution of the reload operation, reorganize the table with the following procedure:

1. Specify only the `unload` statement to reload data into the LOB column structure base table.
2. Specify only the `lobunld` statement to reorganize the table.

*2 Only the RDAREA indicated in a `KFPL00721-I` message becomes empty.

*3 If the table storage RDAREA is different from the index storage RDAREA, an alternative method is to release the table storage RDAREA from shutdown status, re-initialize the index storage RDAREA, and then re-create the index (`-k ixrc`). For details, see the procedures for handling errors during batch index creation in the *HiRDB Version 8 System Operation Guide*.

*4 When synchronization point specification is used, the utility stores as many rows of data as displayed in the `KFPL00800-I` message.

8.13.3 Error when creating an index in batch mode or re-creating an index

Table 8-41 shows the status of the database and recovery method in the event of an error during re-creation or batch creation of an index.

Table 8-41: Status of database and recovery method in the event of error during re-creation or batch creation of an index

-l option	Status/Recovery method		Message output immediately before error occurred			
			None	725	715	716
`a` or `p`	Status		Status before execution	Status before execution	Index not created	Index creation completed
	Recovery		Re-execute re-creation	Re-execute re-creation	Execute batch creation*	Unnecessary
`n`	Status	Table	Status before execution	Status before execution	Status before execution	Status before execution
		Index	Status before execution	Status before execution	Not guaranteed	Creation completed
	Recovery	Table	Unnecessary	Unnecessary	Unnecessary	Unnecessary
		Index	Re-execute re-creation	Re-execute re-creation	Re-execute batch creation after recovering database*	Unnecessary

Legend:

725: Index unloading started

715: Batch index creation started

716: Batch index creation completed

Note

Messages are abbreviated. For example, `725` means `KFPL00725-I`.

* If the index to be recovered is a plug-in index, the utility can be executed only if the batch plug-in index creation partial recovery facility is provided. Otherwise, the index re-creation process is re-executed.

8.13.4 Error when reorganizing an index

Table 8-42 shows the status of the database and recovery method in the event of an error during index reorganization.

Table 8-42: Status of database and recovery method in the event of error during index reorganization

-l option	Status/Recovery method		Message output immediately before error occurred			
			None	725	715	716
a or p	Status		Status before execution	Status before execution	Index not created	Index creation completed
	Recovery		Re-execute reorganization	Re-execute reorganization	Execute batch creation*	Unnecessary
n	Status	Table	Status before execution	Status before execution	Status before execution	Status before execution
		Index	Status before execution	Status before execution	Not guaranteed	Creation completed
	Recovery	Table	Unnecessary	Unnecessary	Unnecessary	Unnecessary
		Index	Re-execute reorganization	Re-execute reorganization	Re-execute batch creation after recovering database*	Unnecessary

Legend:

725: Index unloading started

715: Batch index creation started

716: Batch index creation completed

Note

Messages are abbreviated. For example, `725` means `KFPL00725-I`.

* If you recovered the index storage RDAREA by re-initializing the RDAREA instead of using its backup copy, re-execute the index re-creation.

8.13.5 Error when unloading LOB data

If an error occurs while LOB data is being unloaded, the contents of the database remain unchanged from when the unload operation started. In the event of an error, eliminate the cause of the error and then re-execute the unload operation only on the LOB column. When the LOB column structure base table is reorganized separately from the LOB column, the `unload` and `lobunld` statements are both specified. If an error occurs while LOB data is being unloaded, in order to start processing from the unloading of the LOB data, delete the `unload` statement and re-execute the unload operation with only the `lobunld` statement specified.

8.13.6 Error when reloading LOB data

Table 8-43 describes the database status and recovery method in the event of an error during a LOB data reload operation.

Table 8-43: Database status and recovery method in the event of an error during LOB data reload operation

-l option	Status/Recovery method		Message output immediately before error occurred			
			None	721	712 and 732	714 and 734
a or p	Status	Table or index	Status before execution	Reorganization of LOB column structure base table completed	Reorganization of LOB column structure base table completed	Table reorganization completed
		LOB column	Status before execution	Null	Null	LOB column reorganization completed
	Recovery	Table or index	Unnecessar y	Unnecessary	Unnecessary	Unnecessary
		LOB column	Re-execute reloading of LOB column only	Re-execute reloading of LOB column only	Re-execute reloading of LOB column only	Unnecessary
n	Status	Table or index	No-log shutdown	No-log shutdown	No-log shutdown	Table reorganization completed
		LOB column	Not guaranteed	Not guaranteed	Not guaranteed	LOB column reorganization completed
	Recovery	Table or index	Re-execute after recovering database*	Re-execute after recovering database*	Re-execute after recovering database*	Unnecessary
		LOB column	Re-execute after recovering database*	Re-execute after recovering database*	Re-execute after recovering database*	Unnecessary

Legend:

721: LOB data deleted

712 and 732: Reloading of LOB data started

714 and 734: Reloading of LOB data completed

Note 1

Messages are abbreviated. For example, `712` means `KFPL00712-I` and `732` means `KFPL00732-I`.

Note 2

When both the LOB column structure base table and the LOB column are reorganized without the `-j` option specified, the `unload` and `lobunld` statements are both used. In the event of an error during the unload or reload operation on the LOB column, you must delete the `unload` statement and re-execute the processing with only the `lobunld` statement specified, because reorganization of the LOB column structure base table has been completed. In the event of abnormal termination of the unload operation, you can re-execute `-k rorg` as is; however, in the event of abnormal termination of the reload operation, you must change it to `-k reld`.

* If the reload operation on a LOB column terminates abnormally in the no-log mode, you must recover the table (LOB column structure base table, LOB column, columns of an abstract data type with the LOB attributes, and indexes) from its backup and then re-execute table reorganization.

Chapter

9. Dictionary Import/Export Utility (pdexp)

This chapter explains the dictionary import/export utility (`pdexp`) that migrates table definition information, trigger definition information, and stored procedure information between HiRDB systems.

This chapter contains the following sections:

9.1 Overview

Executor: User with DBA privilege

The dictionary import/export utility migrates table definition information, trigger definition information, and stored procedure information between HiRDB systems.

This utility outputs table definition information, trigger definition information, or stored procedure information to an export file and migrates the information by using the export file as the input file to another HiRDB system. Output of table definition information, trigger definition information, or stored procedure information to an export file is called *exporting*; input of the contents of an export file to a HiRDB system is called *importing*. This utility can also generate a definition SQL on the basis of the definition information in export files.

Figure 9-1 provides an overview of the dictionary import/export utility (`pdexp`) (importing and exporting table definition information, trigger definition information, or stored procedure information), and Figure 9-2 provides an overview of generating a definition SQL.

You import or export table definition information, trigger definition information, or stored procedure information in the following cases:

- When the same table, trigger, or stored procedure is to be used by multiple HiRDB systems
- When a test system is to be migrated to the actual system
- When existing tables, triggers, or stored procedures are to be exported and then imported back to the same system for reasons such as system reconstruction

You generate a definition SQL in the following case:

- When the definition SQL for an existing table, trigger, or stored procedure is to be used as the basis for creating a different table, trigger, or stored procedure

Figure 9-1: Overview of dictionary import/export utility (pdexp) (importing and exporting table definition information, trigger definition information, or stored procedure information)

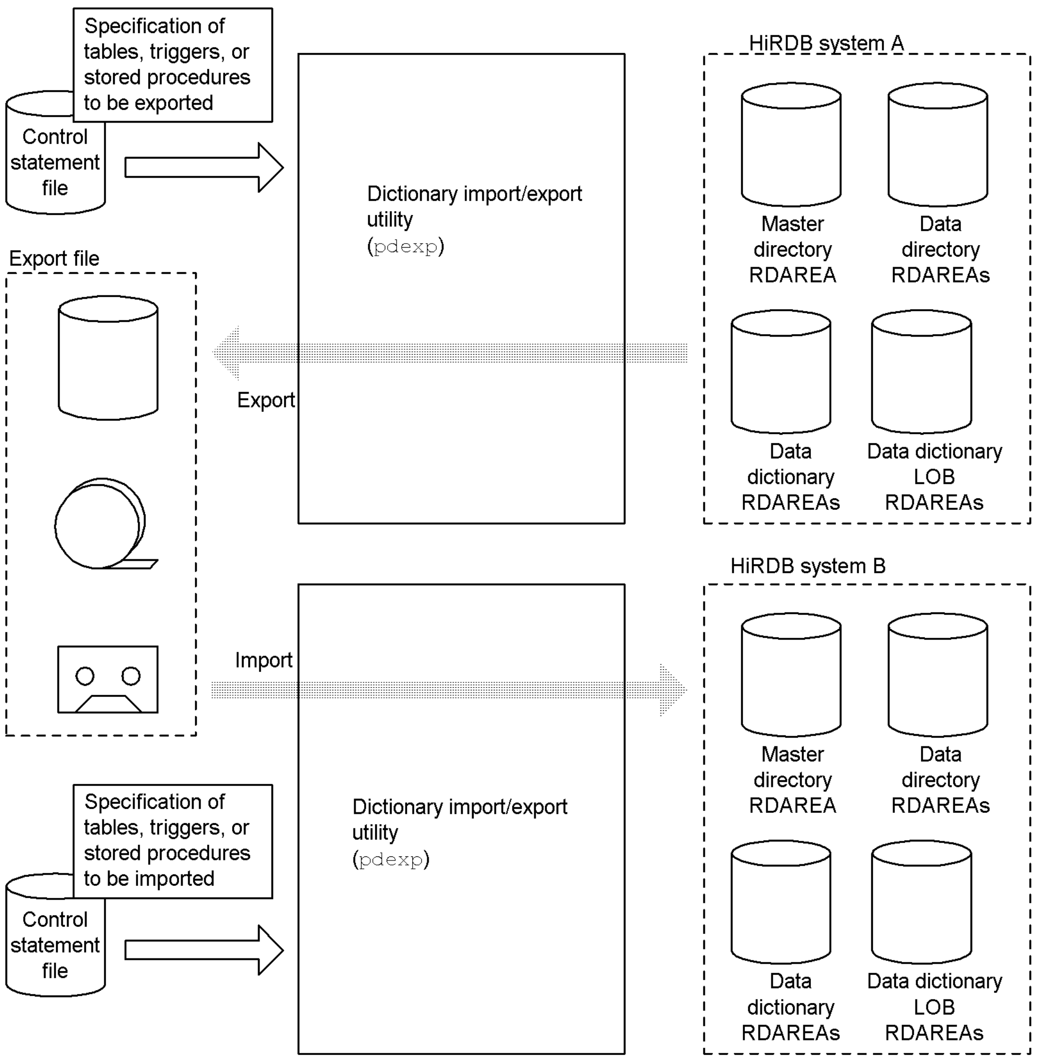
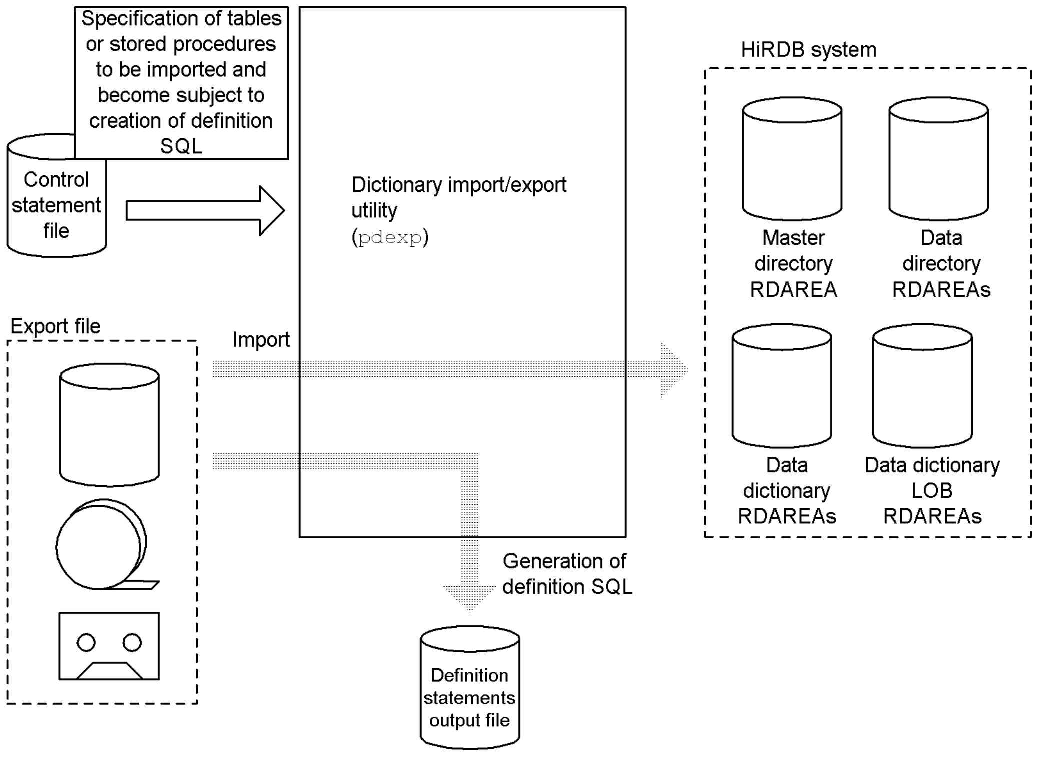

Specification of tables, triggers, or stored procedures to be exported
Control statement file
Dictionary import/export utility (pdexp)
HiRDB system A
Master directory RDAREA
Data directory RDAREAs
Data dictionary RDAREAs
Data dictionary LOB RDAREAs
Export file
Export
Import
HiRDB system B
Master directory RDAREA
Data directory RDAREAs
Data dictionary RDAREAs
Data dictionary LOB RDAREAs
Specification of tables, triggers, or stored procedures to be imported
Control statement file
Dictionary import/export utility (pdexp)

Figure 9-2: Overview of dictionary import/export utility (pdexp) (generating a definition SQL)


Specification of tables or stored procedures to be imported and become subject to creation of definition SQL
Control statement file
Dictionary import/export utility (pdexp)
HiRDB system
Master directory RDAREA
Data directory RDAREAs
Export file
Import
Data dictionary RDAREAs
Data dictionary LOB RDAREAs
Generation of definition SQL
Definition statements output file

9.1.1 Table definition information import/export

Import/export of table definition information involves migration of table definition information. The table definition information remains in the export source dictionary tables after transfer. The following types of table definition information can be imported and exported:

- Table definition information for base tables
- Definition information for indexes
- Table definition information for view tables
- Base table and view table comment information
- Table alias definition information

Because only the definition information is transferred by table definition information import/export, the database reorganization utility (`pdrorg`) must be used in order to transfer the table data. A definition SQL can also be generated from the export file definition information.

9.1.2 Trigger definition information import/export

Import/export of trigger definition information involves migration of trigger definition information. The trigger definition information remains in the source data dictionary table after migration. You can generate a definition SQL on the basis of the definition information in the export file.

9.1.3 Stored procedure information import/export

Import/export of stored processing information involves migration of the definition source statements of stored procedures. The stored procedure information remains in the source data dictionary table after migration.

The utility exports the definition source statements of a stored procedure during export processing, and HiRDB creates an object on the basis of the definition source statements of the stored procedure during import processing.

You can also generate a definition SQL on the basis of the definition information in the export file.

9.1.4 Generation of a definition SQL

A definition SQL is generated and output to a file on the basis of the export file definition information. The following table lists the definition SQL statements that can be generated:

Type	Index definition	Comment definition	Definition SQL statements to be generated

Base table	Specified	Specified	`CREATE TABLE`
			`CREATE INDEX`
			`COMMENT`

		Not specified	`CREATE TABLE`
			`CREATE INDEX`

	Not specified	Specified	`CREATE TABLE`
			`COMMENT`

		Not specified	`CREATE TABLE`

View table	--	Specified	`CREATE VIEW`
			`COMMENT`

		Not specified	`CREATE VIEW`
Alias table	--	--	`CREATE ALIAS`
Trigger	--	--	`CREATE TRIGGER`

Type	Index definition	Comment definition	Definition SQL statements to be generated
Stored procedure	—	—	`CREATE PROCEDURE`

Legend: — : Not applicable

9.2 Command format

9.2.1 Format

Import and export of table definition information, trigger definition information, or stored procedure information:

```
pdexp{-e export-filename| -i export-filename}
     [-f control-statements-filename][-u authorization-identifier]
```

Generation of definition SQL:

```
pdexp -i export-filename [-f control-statements-filename]
      -o definition-SQL-output-filename [-a] [-u authorization-identifier]
```

9.2.2 Options

- `-e` *export-filename* ~ <[identifier:] pathname> ((up to 1023 bytes))

For exporting table definition information, trigger definition information, or stored procedure information:

Specifies a name for the export file.

For generating a definition SQL:

Specifies the name of the export file used at the time of export of the table definition information, trigger definition information, or stored procedure information that is to be created.

A colon and the absolute pathname of the host that contains the export file must also be specified. If the identifier is omitted, the host name of the server machine where the dictionary import/export utility is executed (server machine where the pdexp command is entered) is assumed. If the specified export file is not found, a new file belonging to the user who started the HiRDB is created.

- `-i` *export-filename* ~ <[identifier:] pathname> ((up to 1023 bytes))

Specifies the name of the export file for importing table definition information, trigger definition information, or stored procedure information or for generating a definition SQL.

A colon and the absolute pathname of the host that contains the export file must also be specified. If the identifier is omitted, the host name of the server machine where the dictionary import/export utility is executed (server machine where the `pdexp` command is entered) is assumed.

- `-f` *control-statements-filename* ~ <pathname> ((up to 1023 bytes))

Specifies the name of the file that contains the dictionary import/export utility control statements.

Create the control statement file on the server machine where the dictionary import/export utility will be executed (server machine where the `pdexp` command will be entered). You can omit this specification if all table definition information, trigger definition information, or stored procedure information in the export file is to be imported or if all definition information in the export file is to be used to generate a definition SQL.

Rules for control statements

1. You can specify a maximum of 64 tables, triggers, or stored procedures for import or export.
2. Only one item can be specified per line.
3. A single specification cannot span multiple lines.
4. Null lines are ignored.
5. `PUBLIC`, `MASTER`, `ALL`, and `HiRDB` cannot be used as schema names. (You can specify `PUBLIC` for a table.)
6. Specifications of tables (`-t`), triggers (`-g`), and stored procedures (`-p`) cannot be intermixed.

Importing or exporting table definition information or generating a definition SQL:

```
 -t schema-name.{table-identifier|table-name} [-w]
[-t schema-name.{table-identifier|table-name} [-w]]
```

Importing or exporting trigger definition information or generating a definition SQL:

```
 -g schema-name.trigger-identifier [-w]
[-g schema-name.trigger-identifier [-w]]
```

Importing or exporting stored procedure information or generating a definition SQL:

```
 -p schema-name.routine-identifier
[-p schema-name.routine-identifier]
```

`-t` *schema-name*`.`{*table-identifier*|*table-name*}

Specifies a table that is to be imported, exported, or subject to definition SQL generation.

The same table cannot be specified more than once. To specify both base and view tables in a single control statement file, specify the base tables first.

External tables cannot be specified. If an external table is specified, the utility skips its processing.

To import or export a public view, specify `PUBLIC` as the schema name.

schema-name ~ <identifier> ((1-30))

Specifies the schema name. If a schema name is enclosed in double quotation marks ("), it is handled as case sensitive; otherwise, it is handled as all uppercase letters.

{*table-identifier*|*table-name*} ~ <identifier> ((1-30))

Specifies a table identifier or table name. If a table identifier or table name is enclosed in double quotation marks ("), it is treated as being case sensitive; otherwise, it is treated as in all uppercase letters. If a table identifier or table name contains a space, enclose the entire table identifier or table name in double quotation marks (").

`-w`

Specifies that the table definition of a referencing table for which `WITH PROGRAM` was not specified is to be imported as the table definition of a referencing table for which `WITH PROGRAM` is specified. When the `-w` option is omitted or the specified table definition is for a table other than a referencing table, the system imports its table definition as a definition in which `WITH PROGRAM` is specified. This option is ignored during export processing.

`-g` *schema-name*`.`*trigger-identifier*`[-w]`

Specifies a trigger that is to be imported, exported, or subject to definition SQL generation.

The same trigger cannot be specified more than once.

schema-name ~ <identifier> ((1-30))

Specifies a schema name. If a schema name is enclosed in double quotation marks ("), it is treated as being case-sensitive; otherwise, it is treated as in all uppercase letters.

trigger-identifier ~ <identifier> ((1-30))

Specifies a trigger identifier. If a trigger identifier is enclosed in double quotation marks ("), it is treated as being case-sensitive; otherwise, it is treated as in all uppercase letters. If a trigger identifier contains a space, enclose the entire trigger identifier in double quotation marks (").

`-w`

Specifies that a trigger definition without `WITH PROGRAM` specified is to be imported to a trigger definition with `WITH PROGRAM` specified. If the `-w` option is specified to import a trigger definition with `WITH PROGRAM` specified, the system ignores any existing SQL object whose functions and procedures for the target table are in effect.

Notes when multiple triggers are defined for a single table

If multiple triggers are defined for a single table, the triggers function in the order they are defined. If the order of trigger definitions at the time of import processing is different from the order at the time of export processing, the order in which the triggers function will be different before and after migration. To achieve the same trigger operations before and after migration, you must specify the trigger names in the control statements in the order they are defined.

The following describes how to create control statements in the order of the trigger definitions:

When importing or exporting all triggers defined for a specific table:

The following SQL statement obtains the trigger owner, trigger identifier, and trigger definition time of each trigger defined for table in the order of the trigger definitions:

```
SELECT TRIGGER_SCHEMA,TRIGGER_NAME,CREATE_TIME
  FROM "MASTER".SQL_TRIGGERS
  WHERE TABLE_SCHEMA='owner' AND TABLE_NAME='table-name'
  ORDER BY CREATE_TIME WITHOUT LOCK NOWAIT;
```

In the control statement, specify the schema names and trigger identifiers in chronological order of the trigger definition times.

When importing or exporting all defined triggers:

The following SQL statement obtains the trigger owner, trigger identifier, and trigger definition time of each trigger defined for tables in the order of the trigger definitions:

```
SELECT TRIGGER_SCHEMA,TRIGGER_NAME,CREATE_TIME
  FROM "MASTER".SQL_TRIGGERS
  ORDER BY CREATE_TIME WITHOUT LOCK NOWAIT;
```

In the control statement, specify the schema names and trigger identifiers in chronological order of the trigger definition times.

`-p` *schema-name.routine-identifier*

Specifies a stored procedure to be imported, exported, or subject to definition SQL generation.

The same stored procedure cannot be specified more than once.

schema-name ~ <identifier> ((1-30))

Specifies a schema name. If a schema name is enclosed in double quotation marks (`"`), it is treated as being case-sensitive; otherwise, it is treated as in all uppercase letters.

routine-identifier ~ <identifier> ((1-30))

Specifies a routine identifier. If a routine identifier is enclosed in double quotation marks, it is handled as case sensitive; otherwise, it is handled as all uppercase letters. A routine identifier must be enclosed in double quotation marks if it includes a blank.

■ `-u` *authorization-identifier* ~ <identifier> ((1-8))

Specifies the authorization identifier of the user who will execute the dictionary import/export utility. When this option is specified, a message will be displayed requesting entry of the specified user's password. When the user responds by entering his/her authorization identifier/password, HiRDB checks whether or not that authorization identifier has access privileges to the specified tables; if so, the user is connected to the HiRDB system.

Do not specify this option in an environment in which a response cannot be entered, such as in the background mode where `&` is appended in the shell or in a remote shell.

When this option is omitted, one of the following assumptions is made about the authorization identifier/password:

- The value set in the `PDUSER` environment variable is assumed when the dictionary import/export utility is executed. If you execute the utility in an environment in which a password cannot be entered, such as in the background mode where `&` is appended in the shell or in a remote shell, make sure that you specify `PDUSER`.
- If the `PDUSER` environment variable is not set, the user name entered in the log-in window is assumed. A message will be displayed requesting entry of the password, at which time the password must be entered as the response.

  If an authorization identifier is enclosed in double quotation marks, it is handled as case sensitive; otherwise, it is handled as in all uppercase letters. If you use the Bourne shell (`sh`), C shell (`csh`), or Korn shell (`ksh`), it is also necessary to enclose the authorization identifier in single quotation marks.

- `-o` *definition-SQL-output-filename* ~ <pathname> ((up to 1023 bytes))

Specifies the name of the file to which the generated definition SQL is to be output. If the specified file already exists, it is overwritten; otherwise, it is created.

- `-a`

Specifies that both import and definition SQL generation are to be executed. If this option is specified but the `-o` option omitted, only import is executed.

9.3 Rules and notes

(1) Rules

1. The dictionary import/export utility can be executed only while HiRDB is active. For a HiRDB/Single Server, the single server must already be started; for a HiRDB/Parallel Server, the following servers must be already be started:

   Table definition information import/export

      For export: Front-end server, dictionary server, and all back-end servers

      For import: Front-end server, dictionary server, and object back-end servers

      The master directory RDAREA, data directory RDAREA, and data dictionary RDAREAs must be placed in open status and released from shutdown status when exporting or importing is executed.

   Stored procedure information and trigger information import/export

      For export: Front-end server and dictionary server

      For import: Front-end server and dictionary server

      For import, data dictionary LOB RDAREAs must be defined in advance.

2. The dictionary import/export utility must be executed at the server machine containing the single server or the server machine where the system manager is located.
3. After the dictionary import/export utility has executed, a message is displayed requesting entry of the user's password; the password must then be entered (the entered password is not displayed). A password must be enclosed in double quotation marks if it includes lowercase letters.
4. For the status of RDAREAs associated with the execution of the dictionary import/export utility, see Appendix *C. RDAREA Status During Command Execution*.
5. To execute the dictionary import/export utility, set the `LANG` environment variable. To use character codes that are not supported by the OS in an environment in which the dictionary import/export utility is executed, you must set the `PDLANG` environment variable. For details about `LANG` and `PDLANG`, see the *HiRDB Version 8 UAP Development Guide*.
6. If the specified export file is not found during export processing, the utility creates a new file belonging to the user who started the HiRDB. For this reason, the user who starts the HiRDB must have file creation privilege.
7. During import processing, specify an export file that was created successfully

during export processing. The HiRDB administrator must have referencing privilege for that export file.

8. For a HiRDB/Parallel Server, the correspondences between servers and RDAREAs must be the same after importing as they were before exporting.

9. Import/export processing cannot be performed between systems with a different magnitude of byte size or between systems with different character organizations.

10. If you are using the SQL reserved word deletion facility, use the command execution window to set the `PDDELRSVWDFILE` client environment definition, and then execute the dictionary import/export utility. If the value of `PDDELRSVWDFILE` exceeds 8 (bytes), the `PDDELRSVWDFILE` setting is ignored. Furthermore, if the following conditions are not satisfied, an SQL syntax error may occur during import processing:

    - The value of `PDDELRSVWDFILE` in the client environment definition matches the value specified when the resources such as tables were defined.
    - The exporting and importing systems both have the same `pd_delete_reserved_word_file` operand value in the system common definition.
    - The exporting and importing systems both have the same files under `$PDDIR/conf/pdrsvwd`.

(2) Rules for importing and exporting table definition information, trigger definition information, or stored procedure information

1. Importing and exporting cannot both be specified in the same execution of the utility.

2. Table definition information, trigger definition information, or stored procedure information is exported in the order it is specified in the `-t`, `-g`, or `-p` option in the control statement file. The import order is the same as the order in which the same information was exported, regardless of the specification of the `-t`, `-g`, or `-p` option. Import or export processing is completed for each individual table, trigger, or stored procedure before going on to the next table or stored procedure.

3. If there exists at the import destination the same table (a table with the same *authorization-identifier*`.`*table-identifier*), the same trigger (a trigger with the same *authorization-identifier*`.`*trigger-identifier*), or the same stored procedure (a stored procedure with the same *authorization-identifier*`.`*routine-identifier*), an import error results for that table, trigger, or stored procedure.

4. Before starting import processing, make sure that the import destination system has been initialized and that schemas have been defined for the tables, triggers, or stored procedures to be imported.

5. The dictionary import/export utility transfers table definition information from

the export source to the import destination as is. Therefore, RDAREAs for storing the tables must have already been defined at the import destination with the same names as at the export source.

6. The owner of a table, trigger, or stored procedure to be imported must have usage privileges for the RDAREAs that will store the table, trigger, or stored procedure.
7. To import a stored procedure, the tables to be used by the stored procedure must have already been defined at the import destination. If a required table is not defined, the stored procedure cannot be imported.
8. The `-t`, `-g`, and `-p` options cannot all be specified in the same control statement file.
9. Import processing is disabled if an attempt is made to execute an SQL statement whose size exceeds 2 megabytes (size of an SQL statement defining tables, triggers, or stored procedures exceeding 2 megabytes).
10. Definition information and stored procedure information about any of the following tables cannot be exported:
    - Table definitions containing an abstract data type (`CREATE TABLE`)
    - Index definitions specifying an index type (`CREATE INDEX`)
    - Function definitions (`CREATE FUNCTION` and system-defined scalar function)
    - Abstract data type definitions (`CREATE TYPE`)
    - Stored procedures containing an abstract data type (`CREATE PROCEDURE`)
    - Stored procedures that are declared in an abstract data type (`CREATE TYPE`)
    - Procedure calling another procedure from within a procedure definition (`CREATE PROCEDURE`)
    - Procedure calling a function definition (`CREATE FUNCTION` and system-defined scalar function) from within a procedure definition (`CREATE PROCEDURE`)
    - Comments that have an abstract data type column in the table specified in `TABLE` (`COMMENT`)
    - Comments for which the column specified in `COLUMN` is an abstract data type (`COMMENT`)
11. When a shared table is imported using a HiRDB/Parallel Server, all the RDAREAs storing the shared table and shared indexes defined for the shared table in the target environment must be shared RDAREAs. If any are not shared RDAREAs, an SQL error occurs. If a non-shared table is to be imported but a table storage RDAREA is a shared RDAREA, an SQL error occurs. The table

below describes the relationship between shared tables and target RDAREAs. This relationship does not apply to shared tables for a HiRDB/Single Server because shared RDAREAs are not created for a HiRDB/Single Server.

Type of source table	RDAREA definition in the target system		Can be defined?		pdexp processing after error[*]
	Table storage RDAREA	RDAREA storing index defined for the table	Table	Index	
Shared table	Shared RDAREA	Shared RDAREA	Y	Y	Not applicable.
	Shared RDAREA	Not a shared RDAREA	Y	E	Cancels processing and ignores the table definition.
	Not a shared RDAREA	Shared RDAREA	E	N	Cancels processing.
	Not a shared RDAREA	Not a shared RDAREA	E	N	
Non-shared table	Shared RDAREA	Shared RDAREA	E	N	
	Shared RDAREA	Not a shared RDAREA	E	N	
	Not a shared RDAREA	Shared RDAREA	Y	E	Cancels processing and ignores the table definition.
	Not a shared RDAREA	Not a shared RDAREA	Y	Y	Not applicable.

Legend:

Y: Yes, can be defined.

E: Yes, can be defined, but an SQL error results.

N: No, cannot be defined.

* `pdexp` suppresses internal `COMMIT` of the definition SQL statement and performs rollback if an error occurs during processing.

12. A trigger created internally during definition of the referencing table cannot be exported. If such a trigger is specified during export processing, the utility displays the `KFPX28504-W` message (in which case the utility skips the corresponding trigger and exports the next trigger).

(3) Rules for definition SQL generation

1. A definition SQL statement is not generated during export processing (`-e` option specified) even if the `-o` option is specified.

2. Definition SQL statements are generated in the order they are exported.
3. Definition SQL statements for the following tables, procedures, and functions are not generated:
   - Table definitions containing an abstract data type (`CREATE TABLE`)
   - Procedures specifying an abstract data type as a data type for a procedure parameter (`CREATE PROCEDURE`)
   - Procedures within a procedure definition (`CREATE PROCEDURE`)
   - Procedures and functions that are declared in an abstract data type (`CREATE TYPE`)
   - Index definitions specifying an index data type (`CREATE INDEX`)
   - Function definitions (`CREATE FUNCTION`)
4. A semicolon (`;`) is added at the end of each line of a generated definition SQL statement so that the definition SQL can be used as an input to the database definition utility. In the case of `CREATE PROCEDURE` and `CREATE TRIGGER`, `end_proc;` is added.
5. When `CREATE PROCEDURE` and `CREATE TRIGGER` are to be created, the values specified for the SQL optimization option and SQL extension optimizing option are converted to numeric values.
6. When a definition SQL is to be created for a table for which a foreign key or check constraint has been defined, the utility creates a definition SQL statement for the foreign key or check constraint together with the definition SQL statement for the table. When a definition SQL statement for a foreign key or check constraint is created, the location where the foreign key or check constraint name is defined is determined by the value of `PDCNSTRNTNAME` in the client environment definition and the `pd_constraint_name` operand value in the system definition. For details about `PDCNSTRNTNAME`, see the *HiRDB Version 8 UAP Development Guide*; for details about the `pd_constraint_name` operand, see the manual *HiRDB Version 8 System Definition*.
7. The following table describes the linefeed conditions for definition SQL statements that are generated:

Generated definition SQL statement	Linefeed condition
CREATE TABLE	• Between `CREATE` and the table identifier • At each column definition • At each table storage RDAREA • At each table option • At each foreign key • At each check constraint definition • At `WITH PROGRAM` • At every 80th byte, in the case of a row that exceeds 80 bytes
CREATE INDEX	• Between `CREATE` and the index identifier • Between `ON` and the table identifier • At each column definition • At each index storage RDAREA • At each index option • At every 80th byte, in the case of a row that exceeds 80 bytes
CREATE ALIAS	At every 80th byte, in the case of a row that exceeds 80 bytes
COMMENT	At every 80th byte, in the case of a row that exceeds 80 bytes
CREATE VIEW	• Between `CREATE` and `AS` • At every 80th byte, in the case of a row that exceeds 80 bytes
CREATE PROCEDURE	• Between `CREATE` and a parameter mode specification • At a linefeed code in a definition source statement • At the SQL compile option • At an SQL procedure statement • At every 80th byte, in the case of a row that exceeds 80 bytes
CREATE TRIGGER	• Between `CREATE` and the trigger operation • At a linefeed code in a definition source statement • At the SQL compile option • At WITH PROGRAM • At every 80th byte, in the case of a row that exceeds 80 bytes

(4) Rules for importing and exporting tables for which referential constraints are defined

(a) Exporting

1. When a referencing table and a referenced table are exported, the definitions of their primary key and foreign key are also exported unconditionally.

2. If you are exporting both a referencing table and a referenced table at the same time, specify the referenced table first in the control statement (because tables are imported in the order they were exported).

(b) Importing

1. When a referencing table and a referenced table are imported, the definitions of their primary key and foreign key are also imported unconditionally.
2. When a referencing table is imported, the referenced table must have been defined. If an attempt is made to import a referencing table for which no referenced table has been defined, `CREATE TABLE` results in an SQL error. In such a case, the utility terminates import processing without processing the next table.
3. If the import target contains a constraint with the same name, `CREATE TABLE` results in an SQL error. In such a case, the utility terminates import processing without processing the next table.
4. For the primary and cluster keys, new index and table numbers are assigned in the target system.
5. The index identifier for the primary key is determined by the table number used during import processing. Therefore, it may not be the same as for the source system.
6. The following table describes the handling of SQL objects that use a referenced table when a referencing table is imported:

Referencing action during table definition	-w option	Handling of SQL object that uses referenced table	Remarks
`CASCADE`	Specified	To maintain data integrity between the referencing table and referenced table, a trigger is created internally. Because the SQL object is created at that time, the function, procedure, and triggered SQL object that use the referenced table are ignored.	If the function, procedure, and triggered SQL object that use the referenced table already exist, the SQL object is disabled. Therefore, you must re-create the SQL object after importing the referencing table.
	Omitted	The function, procedure, and triggered SQL object that use the referenced table are ignored. `CREATE TABLE` results in an SQL error because data integrity cannot be maintained between the referencing table and referenced table unless that SQL object is re-created.	

Referencing action during table definition	-w option	Handling of SQL object that uses referenced table	Remarks
`RESTRICT`	Specified	If the function, procedure, and triggered SQL object that use the referenced table are enabled, data integrity is lost because rows are updated or deleted in the referenced table. Therefore, the SQL object is disabled.	If the function, procedure, and triggered SQL object that use the referenced table already exist, the SQL object is disabled. Therefore, you must re-create the SQL object after importing the referencing table.
	Omitted	The function, procedure, and triggered SQL object that use the referenced table are ignored. `CREATE TABLE` results in an SQL error because data integrity cannot be maintained between the referencing table and referenced table unless that SQL object is re-created.	

(5) Notes

1. The results of the dictionary import/export utility can be checked on the basis of the return value set by the utility and by whether or not there are any error messages. The utility returns one of the following values:

   `0`: Terminated normally

   `4`: Warning error occurred

   `8`: Some processing terminated normally, but some processing was skipped

   `12`: Terminated without completing any processing
2. After importing, the table access privilege must be set to `GRANT` in the definition SQL.
3. When a view table is to be imported, the corresponding base table (or tables) must have already been imported or defined and access privileges set.
4. Multiple instances of the dictionary import/export utility cannot be executed simultaneously. If the utility is already executing, it cannot be executed again until the current execution has terminated.
5. Import processing is functionally the same as executing the following definition SQL statements at the import destination system (the specification of each SQL should be checked before import processing is executed):

   Table definition information import/export

   - `ALTER TABLE`

- `COMMENT`
- `CREATE ALIAS`
- `CREATE INDEX`
- `CREATE TABLE`
- `CREATE VIEW`

Trigger definition information import/export

- `CREATE TRIGGER`

Stored procedure information import/export

- `CREATE PROCEDURE`

For details about the SQL specifications, see the *HiRDB Version 8 SQL Reference*.

6. If export processing is interrupted by forced termination of the utility or a server interrupt, the table definition information, trigger definition information, or stored procedure information may not have been exported to the export file, even if the `KFPX28402-I` message is displayed. In such a case, re-execute the dictionary import/export utility and terminate export processing normally.
7. To execute `pdexp`, access privileges for the export file and definition SQL output file must have been granted to the HiRDB administrator.

(6) Example of correcting errors when a definition SQL statement created during import processing exceeds 2 megabytes

When importing data, `pdexp` creates a definition SQL statement and then executes the statement. It is possible that the size of the SQL statement will exceed the permitted maximum length of 2 megabytes, because syntax for default values is also created for the definition SQL statement.

The following example corrects `CREATE TABLE` because the size of this definition SQL statement exceeds 2 megabytes.

Example of CREATE TABLE whose size exceeds 2 megabytes:

When `CREATE TABLE` exceeds 2 megabytes, the utility deletes the column definitions starting with the largest column ID so as to keep its size within 2 megabytes. To add the deleted column definitions, the utility creates `ALTER TABLE`. The following shows a coding example:

```
CREATE TABLE "root"."T1"
("C1" INTEGER NOT NULL,
"C2" CHAR(100),
     :
     :
```

```
"C98" VARCHAR(100)
) IN (("RDUSER01") "C100" < 1500
("RDUSER02"))
PCTFREE=(30,10)
FOREIGN KEY ("C1","C2") REFERENCES "root"."T2"
  ON DELETE RESTRICT ON UPDATE RESTRICT CONSTRAINT CNST01
;    ..............................................1
ALTER TABLE "root"."T1"
ADD "C99" INTEGER
;    ...............................................2
ALTER TABLE "root"."T1"
ADD "C100" INTEGER NOT NULL
;    ...............................................2
```

Explanation

1. `CREATE TABLE` is created without columns `C99` and `C100`.
2. `ALTER TABLE` is created to add columns `C99` and `C100`, which were deleted from `CREATE TABLE`.

SQL error resulting from deleting a column definition in `CREATE TABLE`:

An SQL error occurs if a column deleted in 1 above is any of the following:

- Partitioning key
- Cluster key
- Column specified in a referential constraint
- Column specified in a check constraint search condition

In this example, `CREATE TABLE` results in an SQL error because the partitioning key `C100` has been deleted.

How to avoid SQL errors:

You can use the following procedure to avoid SQL errors:

1. Specify the `-o` option during import processing and create the definition SQL statement (output the definition SQL to a file).
2. Delete the default value section from the `CREATE TABLE` definition SQL statement that was output in step 1.
3. In the definition SQL statement output in step 1, move the first `ALTER TABLE` through the corresponding `ALTER TABLE` (the column definitions that result in the SQL error because they were deleted from `CREATE TABLE`) to the column definitions of `CREATE TABLE`.
4. Use `pddef` to execute the edited definition SQL statement.

The following show an example of editing the above definition SQL statement:

```
CREATE TABLE "root"."T1"
("C1" INTEGER NOT NULL,
"C2" CHAR(100),
     :
     :
"C98" VARCHAR(100)
"C99" INTEGER   ...............Added in step 3
"C100" INTEGER NOT NULL    .....Added in step 3
) IN (("RDUSER01") "C100" < 1500
("RDUSER02"))
PCTFREE=(30,10)
FOREIGN KEY ("C1","C2") REFERENCES "root"."T2"
  CONSTRAINT CNST01    .........Partially deleted in step 2
;
```

(7) Using the files with a BOM

If you selected `utf-8` as the character encoding in the `pdsetup` command, you may be able to use a file with a BOM as the input file for `pdexp`. Table 9-1 shows whether or not files with a BOM can be used with `pdexp`. Note that even when a file with a BOM is used as the input file for `pdexp`, the BOM is skipped. No BOM is included in the file that is output by `pdexp`.

Table 9-1: Whether or not files with a BOM can be used in pdexp (applicable to UTF-8)

Option	Input file	Use of file with a BOM
-l	Export file	N
-f	Control statements file	Y

Legend:

Y: Can be used

N: Cannot be used

9.4 Examples

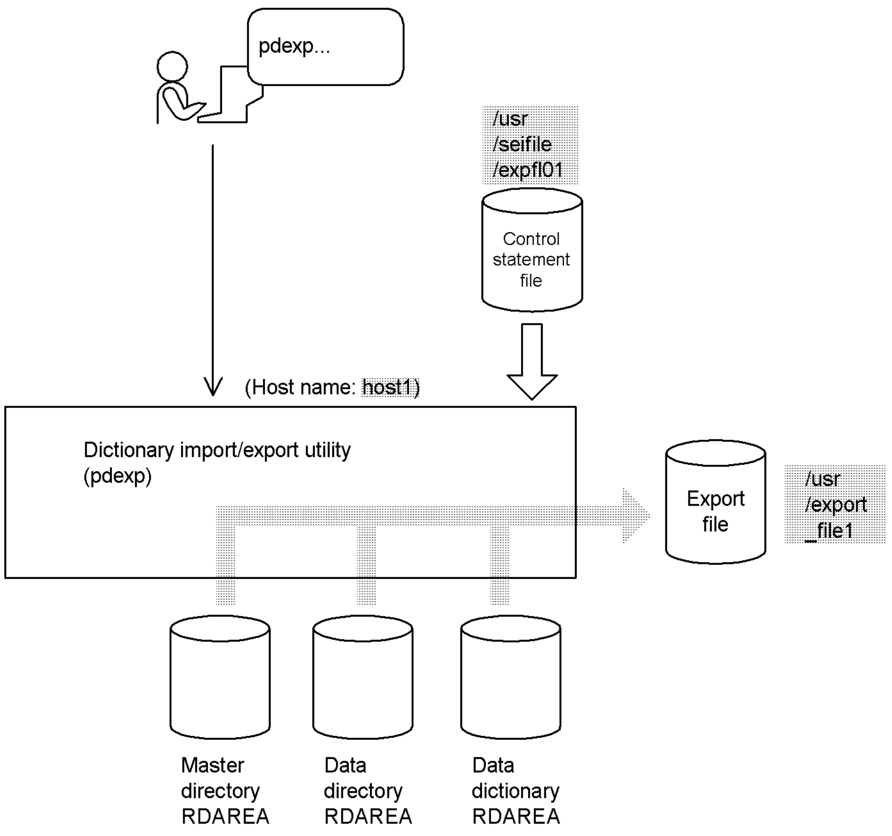
Examples 1-4 show examples of the use of the dictionary import/export utility.

Example 1

Export the definition information for the following tables (schema name: `USER01`):

- `TABLE1`
- `TABLE2`

Overview



Command execution

```
pdexp -e host1:/usr/export_file1
      -f /usr/seifile/expfl01
```

Explanation:

Name of export file: `host1:/usr/export_file1`

Name of control statement file: `/usr/seifile/expfl01`

Contents of control statement file (`/usr/seifile/expfl01`)

```
-t USER01.TABLE1
-t USER01.TABLE2
```

Explanation:

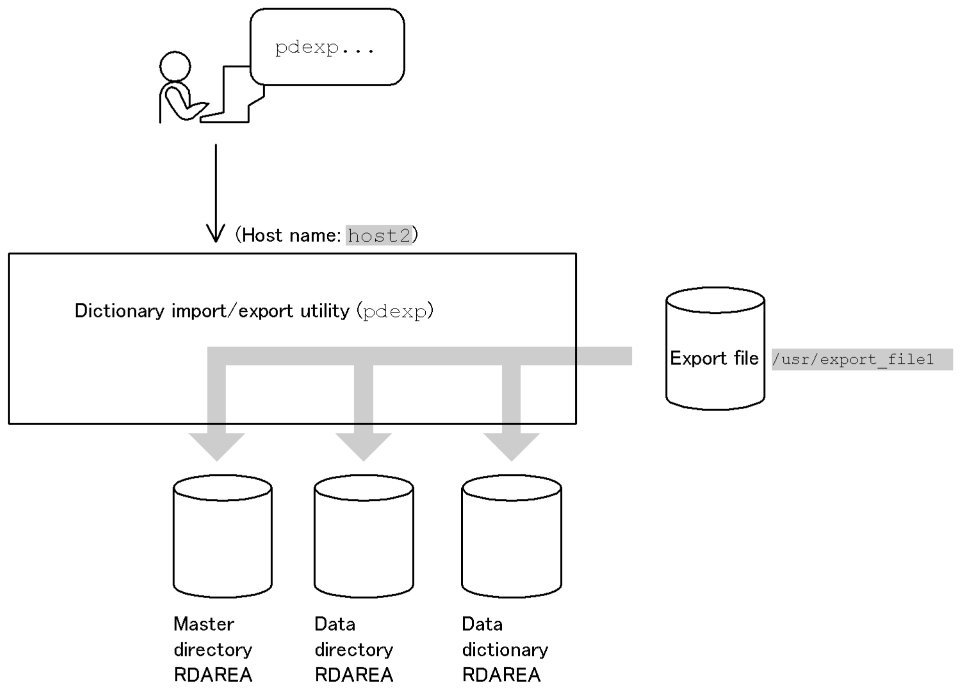
Name of table to which table definition information will be exported: `USER01.TABLE1`

Name of table to which table definition information will be exported: `USER01.TABLE2`

Example 2

Import all table definition information in an export file.

Overview



Command execution

```
pdexp -i host2:/usr/export_file1
```

Explanation:

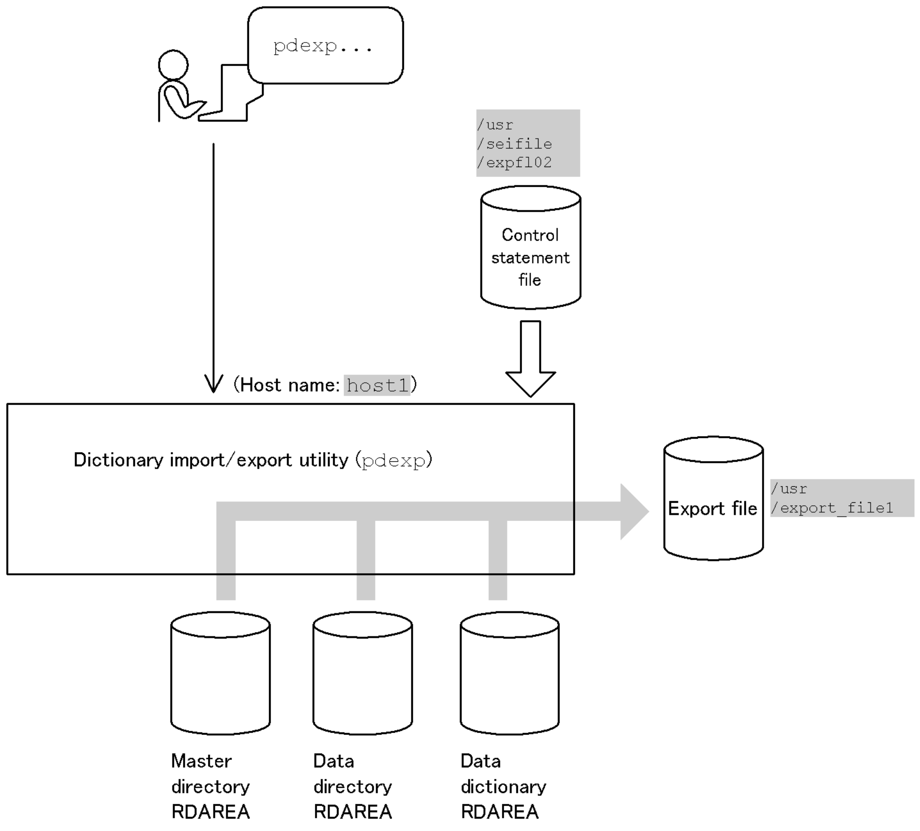
Name of export file: `/usr/export_file1`

Example 3

This example exports definition information about the following stored procedures (schema name: `USER01`):

- `proc1`
- `proc2`

Overview



Command execution

```
pdexp -e host1:/usr/export_file1
      -f /usr/seifile/expfl02
```

Explanation:

Export file: `host1:/usr/export_file1`

Control statement file: `/usr/seifile/expfl01`

Contents of the control statement file (`/usr/seifile/expfl02`**)**

```
-p USER01.PROC1
-p USER01.PROC2
```

Explanation:

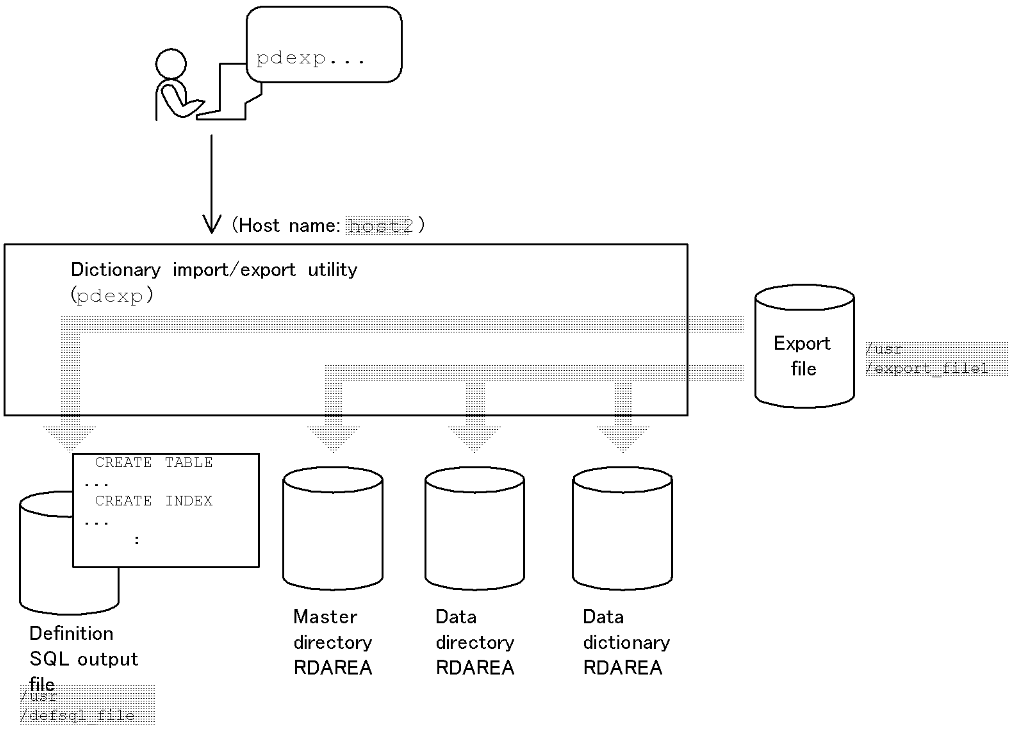
Name of the routine that exports definition information about the stored procedure: `USER01.PROC1`

Name of the routine that exports definition information about the stored procedure: `USER01.PROC2`

Example 4

This example imports all table definition information from an export file. At the same time, it generates definition SQL statements based on the definition information obtained from the export file.

Overview



Command execution

```
pdexp -i host2:/usr/export_file1   .................1
      -o /usr/defsql_file   .......................2
      -a  ......................................3
```

Explanation:

1. Export file: `host2:/usr/export_file1`
2. Definition SQL output file: `/usr/defsql_file`
3. Simultaneous generation of import and definition SQL statements

Chapter 10. Rebalancing Utility (pdrbal)

This chapter explains the rebalancing utility (`pdrbal`) that rebalances a table partitioned by the hash partitioning method. Rebalancing involves addition of new RDAREAs to handle an increase in the amount of data, and rearrangement of data between the existing RDAREAs and added RDAREAs.

This chapter contains the following sections:

10.1 Overview

10.1.1 Functions of the rebalancing utility

If an RDAREA is added to accommodate new data for a hash-partitioned table, an imbalance of data occurs between the existing RDAREAs and the added RDAREA. The rebalancing utility (`pdrbal`) is used to correct such a data imbalance. Correcting an imbalance of data in a hash-partitioned table is called the rebalancing facility for hash row partitioning.

The rebalancing utility rearranges data in units of hash groups. This is called table rebalancing. The table partitioned by a hash function (`HASHA` to `HASHF`) is called a rebalancing table.

Figure 10-1 shows an overview of the rebalancing utility and Figure 10-2 shows an overview of rebalancing.

Figure 10-1: Overview of the rebalancing utility

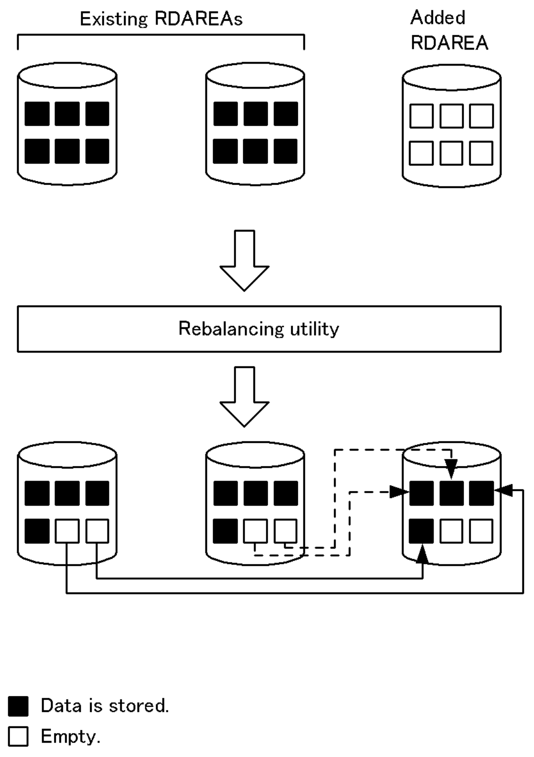
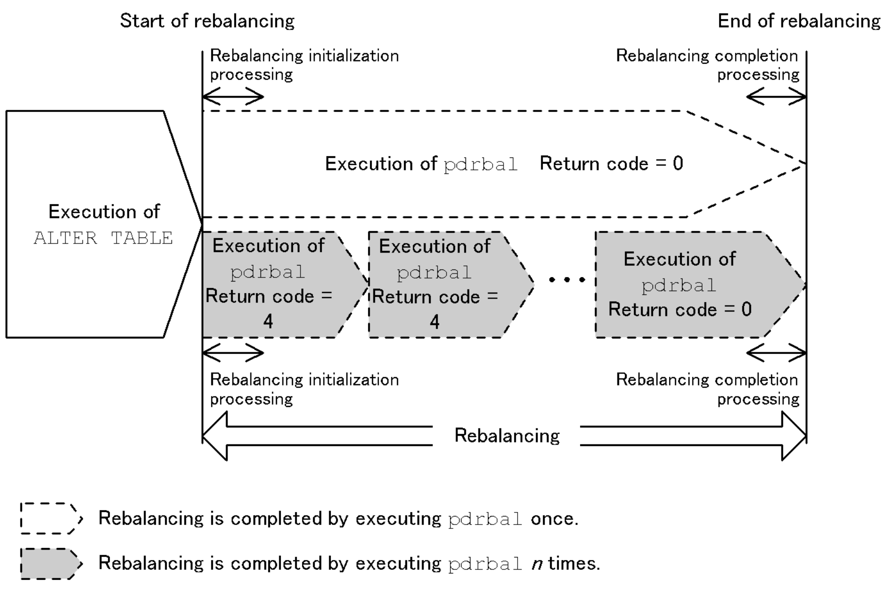


Figure 10-2: Overview of rebalancing


Start of rebalancing
End of rebalancing
Rebalancing initialization processing
Rebalancing completion processing
Execution of ALTER TABLE
Execution of pdrbal Return code = 0
Execution of pdrbal Return code = 4
Execution of pdrbal Return code = 4
Execution of pdrbal Return code = 0
Rebalancing initialization processing
Rebalancing completion processing
Rebalancing
Rebalancing is completed by executing pdrbal once.
Rebalancing is completed by executing pdrbal *n* times.

Explanation:

Start of rebalancing

This is the point at which the first `pdrbal` is executed after an RDAREA has been added (`ALTER TABLE ADD RDAREA`) to the rebalancing table.

Rebalancing initialization processing

This is the initialization processing executed at the first execution of `pdrbal` after an RDAREA has been added.

Rebalancing completion processing

This is the completion processing executed when `pdrbal` has terminated with return code `0`.

End of rebalancing

This is the point at which rebalancing of all table data is completed (`pdrbal` terminated with return code `0`).

Rebalancing

This is a period during which rebalancing is underway.

10.1.2 Operation modes of the rebalancing utility

Table 10-1 shows the operation modes of the rebalancing utility.

Table 10-1: Operation modes of the rebalancing utility

Operation mode	Description
Shared mode	Accesses to the table are permitted during execution of `pdrbal`. Use this mode for a system that cannot stop online applications due to 24-hour operation.
Exclusive mode	Accesses to the table are not permitted during execution of `pdrbal`. Use this mode for a system that can deny access to the table and execute batch processing at night.

(1) Shared mode

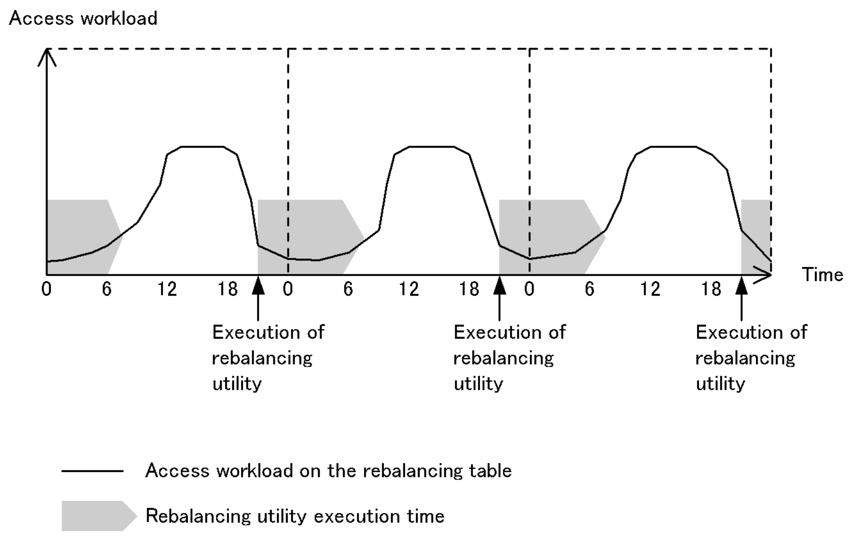
In the shared mode, a table being rebalanced is accessible. To execute the utility in the shared mode, specify `share` in the `-k` option.

The shared mode requires acquisition of log information as well as repeated allocation and release of the resources that are required by concurrently executing SQL. Therefore, the time required for a rebalance operation in the shared mode is longer than in the exclusive mode.

Recommended operation method

When executing the rebalancing utility in the shared mode, you should choose a period of time with little traffic on the rebalancing table. Figure 10-3 shows an example of rebalancing utility execution in the shared mode.

Figure 10-3: Example of rebalancing utility execution in shared mode



(2) Exclusive mode

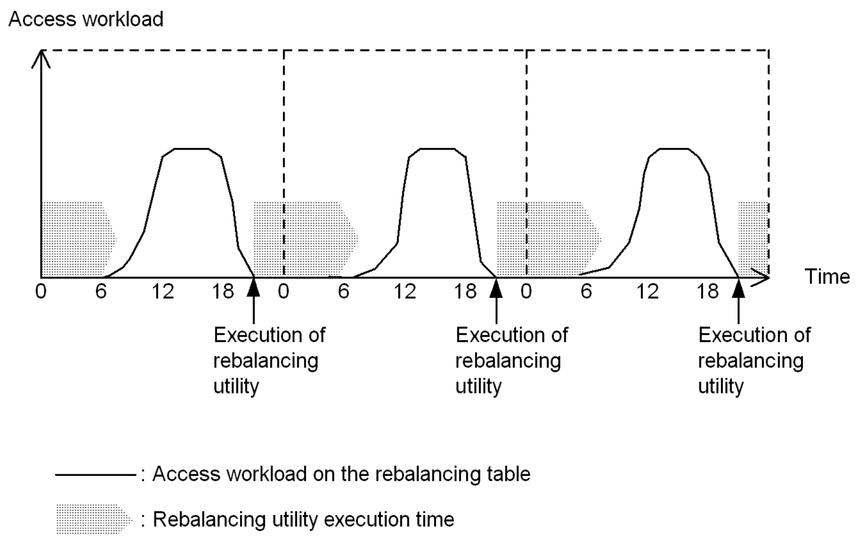
In the exclusive mode, no access is allowed to a table being processed by the rebalancing utility. To execute the rebalancing utility in the exclusive mode, specify `exclusive` in the `-k` option.

In the exclusive mode, acquisition of log information is optional. The required resources are allocated and released only once, and batch index creation is permitted; therefore, the time required for the rebalancing operation is less than in the shared mode.

Recommended operation method

When executing the rebalancing utility in the shared mode, you should choose a period of time during which the rebalancing table is not accessed. Figure 10-4 shows an example of rebalancing utility execution in the exclusive mode.

Figure 10-4: Example of rebalancing utility execution in exclusive mode



10.1.3 Execution environment

1. You can execute the rebalancing utility only when HiRDB is active.
2. You can execute the rebalancing utility at the server machine containing the single server or the server machine where the system manager is located.
3. To execute the rebalancing utility, set the `LANG` environment variable. To use character codes that are not supported by the OS in an environment in which the rebalancing utility is executed, you must set the `PDLANG` environment variable. For details about `LANG` and `PDLANG`, see the *HiRDB Version 8 UAP Development Guide*.
4. The rebalancing utility does not support a utility special unit.

5. Whether or not you can execute the rebalancing utility depends on the open attribute and status of the RDAREAs that contain the rebalancing table and indexes. For details about whether or not the rebalancing utility can be executed, see Appendix *C. RDAREA Status During Command Execution*.

6. When a recovery-unnecessary front-end server is used, whether or not `pdrbal` is executable depends on the applicability of the recovery-unnecessary front-end server and the operating status of the front-end servers, as shown below:

Recovery-unnecessary front-end server		**Whether or not pdrbal is executable**
Front-end servers that have not been applied	**Front-end servers that have been applied**	
All active	None	Y
Some inactive		N (because the front-end server is inactive)
All inactive		N (because the table definition cannot be acquired)
All active	All active	Y
	Some inactive	Y
	All inactive	Y
None	All active	Y
	Some inactive	Y
	All inactive	N (because the table definition cannot be acquired)

Legend:

Y: Executable

N: Not executable

10.1.4 Executors

The following users can execute the rebalancing utility:

- Users with the `DBA` privileges
- Users with the `SELECT`, `INSERT`, and `DELETE` privileges

When the Directory Server linkage facility is used, the system permits execution of the rebalancing utility only when the above privileges are included in the `pdrbal` executor's privileges or in the executor's role privileges.

10.2 Examples

This section presents examples (1-4) of using the rebalancing utility.

(1) Shared mode

Example 1

This example adds an RDAREA (`user03`) to a table (`reb_table`) in a HiRDB/ Single Server and executes the rebalancing utility in the shared mode.

- Table definition:

```
CREATE TABLE reb_table(hkeys INT NOT NULL,names CHAR(30))
    FIX HASH HASHA BY hkeys IN (user01,user02)
```

- Index definition:

```
CREATE INDEX idx1 ON reb_table(hkeys) IN ((idx01),(idx02))
```

■ **Adding an RDAREA with ALTER TABLE**

```
ALTER TABLE reb_table ADD RDAREA user03 FOR INDEX idx1 in idx03
```

Explanation:

This statement adds RDAREA `user03` to table `reb_table`. It also adds RDAREA `idx03` to index `idx1`.

■ **pdrbal command**

```
pdrbal -k share -t reb_table control_file
```

Explanation:

`-k share`: Operation mode (shared mode)

`-t reb_table`: Name of the rebalancing table

`control_file`: Control information file

■ **Contents of the control information file (control_file)**

```
execstop time,9:00    1
report /dsk01/rest_file    2
```

Explanation:

1. Terminates `pdrbal` nine hours later.
2. Outputs the execution result of `pdrbal` to the process results file (`/dsk01/rest_file`).

Example 2

This example adds an RDAREA (`user03`) to a table (`sgml_table`) in a HiRDB/Single Server and executes the rebalancing utility in the shared mode. The `sgml_table` table contains the columns of abstract data type provided by a plug-in.

- Table definition:

```
CREATE TABLE sgml_table (
    hkeys INT NOT NULL,
    doctext SGMLTEXT ALLOCATE (sgmltext IN ((LOB1D),
                               (LOB2D)))
    PLUGIN '<DTD>NEWSPAPER</DTD><EXTRACTparm>extract.prm
                     </EXTRACTparm>'
    )
    FIX HASH HASHA BY hkeys IN (user01,user02)
```

- Index definition:

```
CREATE INDEX idx1 ON sgml_table(hkeys) IN ((idx01),(idx02))
CREATE INDEX ngram_index USING TYPE MASTER.NGRAM on
sgml_table(doctext)
    in ((LOB1I),(LOB2I))
```

■ **Adding an RDAREA with ALTER TABLE**

```
ALTER TABLE sgml_table ADD RDAREA user03
   FOR COLUMN doctext ALLOCATE(sgmltext IN LOB3D)
   FOR INDEX idx1 in idx03,ngram_index in LOB3I
```

Explanation:

This statement adds RDAREA `user03` to table `sgml_table`. It also adds RDAREA `idx03` to index `idx1` and RDAREA `LOB3I` to plug-in index `ngram_index`.

■ **pdrbal command**

```
pdrbal -k share -t sgml_table control_file
```

Explanation:

`-k share`: Operation mode (shared mode)

`-t sgml_table`: Name of the rebalancing table

`control_file`: Control information file

■ **Contents of the control information file (control_file)**

```
execstop time,12:00    1
unld_func type=sgmltext,func=unsgmltext(sgmltext)    2
reld_func type=sgmltext,func=sgmltext(blob)    3
report /dsk01/rest_file    4
```

Explanation:

1. Terminates `pdrbal` 12 hours later.
2. Specifies a constructor parameter reverse creation function.

   `sgmltext`: Name of the abstract data type

   `unsgmltext`: Name of the constructor parameter reverse creation function

   `sgmltext`: Type of argument
3. Specifies a constructor function.

   `sgmltext`: Name of the abstract data type

   `sgmltext`: Name of the constructor function

   `blob`: Type of argument
4. Outputs the execution result of `pdrbal` to the process results file (`/dsk01/rest_file`).

(2) Exclusive mode

Example 3

This example adds an RDAREA (`user03`) to a table (`reb_table`) in a HiRDB/Single Server and executes the rebalancing utility in the exclusive mode.

- Table definition:

```
CREATE TABLE reb_table(hkeys INT NOT NULL,names CHAR(30))
     FIX HASH HASHA BY hkeys IN (user01,user02)
```

- Index definition:

```
CREATE INDEX idx1 ON reb_table(hkeys) IN ((idx01),(idx02))
```

■ **Adding an RDAREA with ALTER TABLE**

```
ALTER TABLE reb_table ADD RDAREA user03 FOR INDEX idx1 in idx03
```

Explanation:

This statement adds RDAREA `user03` to table `reb_table`. It also adds RDAREA `idx03` to index `idx1`.

■ **pdrbal command**

```
pdrbal -k exclusive -t reb_table -l n control_file
```

Explanation:

`-k exclusive`: Operation mode (exclusive mode)

`-t reb_table`: Name of the rebalancing table

`-l n`: no-log mode

`control_file`: Control information file

■ **Contents of the control information file (control_file)**

```
idxwork /idxwork1   1
sort /sortwork   2
report /dsk01/rest_file   3
```

Explanation:

1. Specifies a directory for index information files (`/idxwork1`).
2. Specifies a work directory for sorting (`/sortwork`).
3. Outputs the execution result of `pdrbal` to the process results file (`/dsk01/rest_file`).

Example 4

This example adds an RDAREA (`user03`) to a table (`reb_table`) in a HiRDB/Parallel Server and executes the rebalancing utility in the exclusive mode.

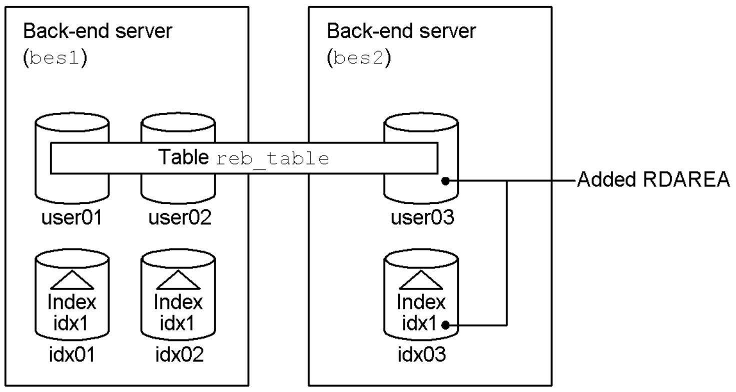
- Table definition:

```
CREATE TABLE reb_table(hkeys INT NOT NULL,names CHAR(30))
    FIX HASH HASHA BY hkeys IN (user01,user02)
```

- Index definition:

```
CREATE INDEX idx1 ON reb_table(hkeys) IN ((idx01),(idx02))
```

■ **Relationship among servers, RDAREAs, table, and index**



■ **Adding an RDAREA with ALTER TABLE**

```
ALTER TABLE reb_table ADD RDAREA user03 FOR INDEX idx1 in idx03
```

Explanation:

This statement adds RDAREA `user03` to table `reb_table`. It also adds RDAREA `idx03` to index `idx1`.

■ **pdrbal command**

```
pdrbal -k exclusive -t reb_table -l n control_file
```

Explanation:

`-k exclusive`: Operation mode (exclusive mode)

`-t reb_table`: Name of the rebalancing table

`-l n`: no-log mode

`control_file`: Control information file

Contents of the control information file (control_file)

```
idxwork bes2 /idxwork_bes2    1
sort bes2 /sortwork_bes2    2
report /dsk01/rest_file    3
```

Explanation:

1. Specifies a directory for index information files (`/idxwork_bes2`) at the `bes2` back-end server.
2. Specifies a work directory for sorting (`/sortwork_bes2`) at the `bes2` back-end server.
3. Outputs the execution result of `pdrbal` to the process results file (`/dsk01/rest_file`).

10.3 Command format

10.3.1 Format

This section explains the format of the `pdrbal` command. In the following table, each number corresponds to the number assigned to each option.

No.	Format
1	`pdrbal [-k` *operation-mode*`]`
2	`-t [`*authorization-identifier*`.]`*table-identifier*
3	`[-i` *index-creation-method*`]`*
4	`[-l` *log-acquisition-method*`]`*
5	`[-u` *authorization-identifier*`]`
6	`[-c` *commit-unit*`]`
7	`[-n [`*batch-input/output-local-buffer-sectors-count*`],` `[`*random-access-local-buffer-sectors-count*`]]`*
8	`[-m` *progress-message-output-interval*`]`
9	`[`*control-information-filename*`]`

Note

If you specify *control-information-filename*, be sure to specify it as the last option.

* This option is ignored if specified in the shared mode (`-k share`).

10.3.2 Options

(1) -k operation-mode

~ <<`share`>>

Specifies `pdrbal`'s operation mode.

`share`

Allows other users to reference and update the table during execution of `pdrbal` (shared mode). Use this mode when executing `pdrbal` without terminating online applications for the table.

`exclusive`

Does not allow other users to reference or update the table during execution of `pdrbal` (exclusive mode). Use this mode when executing `pdrbal` by terminating all online applications for the table.

(2) -t [authorization-identifier.]table-identifier

Specifies the name of the rebalancing table being processed by the rebalancing utility.

When the authorization identifier is omitted, the system assumes the authorization identifier of the user who established the connection with HiRDB.

Rules

1. You cannot specify either of the following tables:
    - View tables
    - Foreign tables
    - Non-rebalancing tables (partitioned by a `HASH` function (`HASHA` to `HASHF`))
2. If an authorization identifier or table identifier is enclosed in double quotation marks (`"`), the command treats it as being case sensitive. If it is not enclosed in double quotation marks (`"`), the command treats it as in all uppercase letters. If you are using `sh` (Bourne shell), `csh` (C shell), or `ksh` (Korn shell), you need to enclose the entire identifier in single quotation marks (`'`).
3. Table 10-2 shows whether or not `pdrbal` can be executed on a rebalancing table containing an abstract data type.

Table 10-2: Whether or not pdrbal can be executed on a rebalancing table containing an abstract data type

Contents of abstract data type		Execution of pdrbal
User-defined abstract data type		N
Abstract data type provided by plug-in	With `BLOB` attribute	Y*
	No `BLOB` attribute	Y

Y: Executable.

N: Not executable.

* With some plug-ins, `pdrbal` is executable only when a constructor reverse creation function for database reorganization is specified.

(3) -i index-creation-method

~ <<c>>

Specifies the index creation method.

In the shared mode, this operand, if specified, is ignored (in which case `s` is assumed).

`c`

Indicates the batch index mode. When this mode is specified, the utility rebalances row data and then creates an index in batch mode.

Criterion:

Specify this option if you want to rebalance row data in the exclusive mode and then create an index for the row data at high speed. Note that if you specify the `execstop` statement, the utility creates an index in the batch mode after the data has been moved; therefore, you may not be able to stop `pdrbal` until index creation for the moved data has been completed.

`s`

Indicates the index update mode. In this mode, the utility updates indexes each time row data is moved.

Criterion:

Specify this option if only a small amount of row data is rebalanced in the exclusive mode or it is impossible to allocate an index information file or work directory for sorting. Also specify this option if you want to make sure that `pdrbal` stops within the execution time specified in the `execstop` statement.

Note

1. When `-i c` is specified, the command creates as many index information files as the *number of indexes* × *number of added table storage RDAREAs*. Because these files are opened simultaneously, the maximum number of files permitted per process may be exceeded. If this is the case, increase the `pd_max_open_fds` operand value in the system definition. If the value of the `pd_max_open_fds` operand is exceeded, check and, if necessary, revise the number of table partitions per server and the number of defined indexes or specify `-i s`.
2. If the `index` and `idxwork` statements are omitted, the system outputs the index information file to the `/tmp` directory using the following naming convention:

   `/tmp/`
   `INDEX-`*index-name*`-`*index-storage-RDAREA-name*`-`*unique-character-strin g*

   If `pdrbal` terminates abnormally, this file is not deleted. If you re-execute `pdrbal`, another index information file is created under a different name.

Because this may result in a space shortage in the `/tmp` directory, you should delete unneeded index information files with the OS's `rm` command.

(4) -l log-acquisition-method

~ <<a>>

Specifies the database update log acquisition method during the execution of `pdrbal`.

If you have specified the exclusive mode, be sure to specify the `-l` option. This option is ignored in the shared mode (in which case `a` is assumed).

`a`

Indicates the log acquisition mode in which the system acquires database update log information required for rollback and rollforward.

Criteria:

Specify this option if only a small amount of row data is rebalanced or if you do not want to make backup copies before and after the execution of `pdrbal`.

When `pdrbal` is executed in the log acquisition mode, there is no need to make backup copies before and after the execution of `pdrbal`, but the performance is lower than in the no-log mode.

`n`

Indicates the no-log mode. The system does not collect database update log information.

Criteria:

Specify this option when a large amount of row data is to be rebalanced.

When `pdrbal` is executed in the no-log mode, the execution time is shorter than in the log acquisition mode. However, you need to make a backup copy before executing `pdrbal` to protect against possible abnormal termination. Because the system does not collect database update log information, you also need to make a backup copy after the execution of `pdrbal`.

If `pdrbal` terminates abnormally, the table storage RDAREAs are placed in error shutdown status. In this case, you need to restore the RDAREAs from their backup copy made prior to the execution of `pdrbal`.

Rules

1. If the RDAREAs can be restored from a previous backup copy, there is no need to make a backup copy prior to the execution of `pdrbal`.
2. In the no-log mode, the system outputs the following amount of `ENQ` log per server instead of the database update log:

`ENQ` log = $(p + q + r) \times T$

p: Number of table storage RDAREAs

q: Number of LOB column (LOB attribute) storage RDAREAs

r: Number of index storage RDAREAs

T: Amount of transaction log (see the `-c` option)

3. For details about how to operate in no-log mode, see the *HiRDB Version 8 System Operation Guide*.
4. If the `execstop` statement is specified to execute `pdrbal` in multiple segments, you need to make a backup copy each time `pdrbal` is executed.
5. If you use Real Time SAN Replication based on the log-only synchronous method, and if you execute `pdrbal` with `-l n` specified at the transaction execution site, you must perform the preparations for log application. For details about the preparations for log application, see the manual *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide*.

(5) -u authorization-identifier

Specifies the authorization identifier of the user executing `pdrbal`.

For the default for this option, see *Default value* as follows.

If this option is specified, the system displays a message requesting an entry of a password. If no password is required, enter null in response to the message.

The system establishes connection with HiRDB using the specified authorization identifier and checks execution privileges.

Criterion

Specify this option to execute `pdrbal` using a different authorization identifier than the one specified in the `PDUSER` environment variable.

Default value

When this option is omitted, the system assumes the authorization identifier and password as follows:

1. The system assumes the value of the `PDUSER` environment variable during the execution of `pdrbal`. Be sure to specify `PDUSER` if you are executing the utility in the background with `&` attached by the shell, or in a remote shell environment when a password cannot be entered.Following are examples of the `PDUSER` environment variable:

   Specifying a password:

`setenv PDUSER '"`*authorization-identifier*`"/"`*password*`"'`

Not specifying a password:

`setenv PDUSER '"`*authorization-identifier*`"'`

2. If the `PDUSER` environment variable is not specified, the system assumes the login window's user name. Enter the password when a message is displayed requesting password entry. If no password is required, enter null in response to the message.

Rules

1. Do not specify this option if you are executing the utility in the background with `&` attached by the shell, or in a remote shell environment where a password cannot be entered.
2. If you enclose an authorization identifier in double quotation marks, the system treats it as case sensitive; otherwise, the system treats it as in all uppercase letters. If you use the Bourne shell (`sh`), C shell (`csh`), or Korn shell (`ksh`), you need to enclose the authorization identifier in single quotation marks (`'`).

(6) -c commit-unit

~ <unsigned integer> ((1-1000000)) <<shared mode: 10, exclusive mode: 100000>>

Specifies the number of rows to be moved before a commit point is reached when rearranging row data during a rebalance operation.

If `0` is specified, the utility does not limit the number of rows that can be moved per transaction (the utility moves rows, treating processing through the end of rebalancing or up to the time specified in the `execstop` statement as one transaction).

With a HiRDB/Parallel Server, a commit may be reached before the specified number of rows depending on the number of rows stored in each RDAREA.

Recommended value

In the shared mode, `pdrbal` locks both source and target RDAREAs when moving data. This means that the source and target RDAREAs are not accessible until `pdrbal` stops moving data (until a commit point is reached). Therefore, in the shared mode, you can reduce the lock release wait time by specifying a small value for the commit unit.

If UAP processing is more important than the rebalance operation, specify a small value; if the rebalance operation is more important than UAP processing, specify a large value.

If an index has been defined for the target table in the exclusive and no-log mode, you can improve performance by specifying `0` as the commit unit (an improvement in performance can be expected because the global buffer updated by index maintenance at the time of commit is flushed only once, thereby also reducing the number of commits). However, the number of locked resources increases according to the increase in the number of rows to be processed, because the processing is performed by a single transaction. Also, the interval during which no synchronization point dump is collected becomes longer.

Notes

1. If there are comparatively many transactions for online applications, you should specify a value of 100 or less.
2. If a small value is specified for the commit unit, the rebalance processing requires a long time. As the number of commit points increases, the amount of output transaction log increases. A transaction log is always output regardless of the specification of the `-l` option, and its amount can be determined by the following formula:
   ```
   Amount of transaction log
           = (1328 + 176 × 3) × (a ÷ b + 1024 ÷ c) (bytes)
   ```
   a: Number of data items being rebalanced

   b: Value of `-c` option

   c: Number of table storage RDAREAs

   Each time a commit point is reached, the system outputs this amount of transaction log to each server's system log file. Each server means each and every front-end server and back-end server that contains the table storage RDAREAs. If the table has an abstract data type, transaction log is also output to the dictionary server.
3. If you specify a large value for the commit unit, a synchronization point dump cannot be collected for a long period of time. If an error occurs when executing the utility concurrently with another UAP, the time required for restart processing increases.
4. You can specify 0 as the commit unit only in the exclusive and no-log mode; otherwise, 0 cannot be specified.

(7) -n [batch-input/output-local-buffer-sectors-count],[random-access-local-buffer-sectors-count]

Specifies that rebalancing is to be executed in the exclusive mode using a local buffer. By specifying this option, you can use a local buffer to access the database, thereby reducing the number of input/output operations by batch input/output operations.

When this option is omitted, the utility uses the global buffer to input/output one page at a time.

batch-input/output-local-buffer-sectors-count ~ <unsigned integer> ((2-4096))

Specifies the number of batch input/output local buffer sectors. The batch input/output local buffer is used for data pages.

We recommend that you specify a value in the range 16-32 as the number of batch input/output local buffer sectors. The guideline is 64 kilobytes/page length.

random-access-local-buffer-sectors-count ~ <unsigned integer> ((4-125000))

Specifies the number of random access local buffer sectors. The random access local buffer is used for index pages.

You should change the combination of the number of batch input/output local buffer sectors and the number of random access local buffer sectors according to the table definition. Table 10-3 shows the recommended `-n` option value.

Table 10-3: Recommended -n option specification (pdrbal)

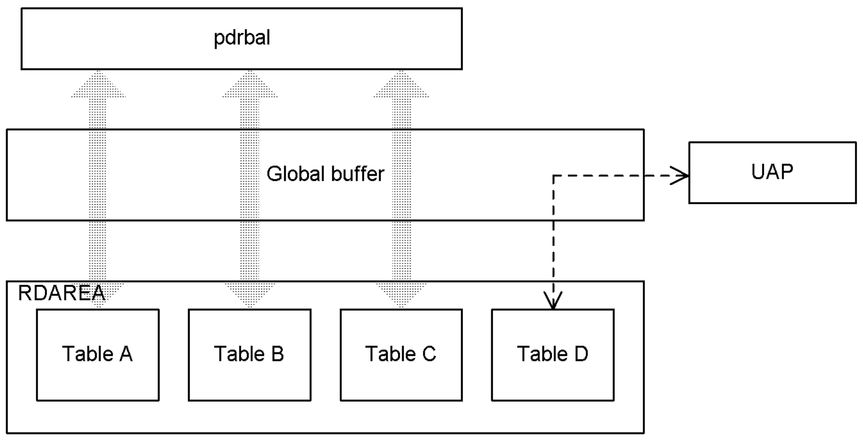
Table type	Column definition	Specification of -n option
FIX table or non-FIX table	All columns are NULL	`-n` *random-access-local-buffer-sectors-count*
Non-FIX table	A variable-length data type column is defined	`-n` *random-access-local-buffer-sectors-count*
	An abstract data type column is defined	
	A `BINARY` column is defined	
FIX table or non-FIX table	Other	`-n` *batch-input/output-local-buffer-sectors-count*

About the buffers used by pdrbal

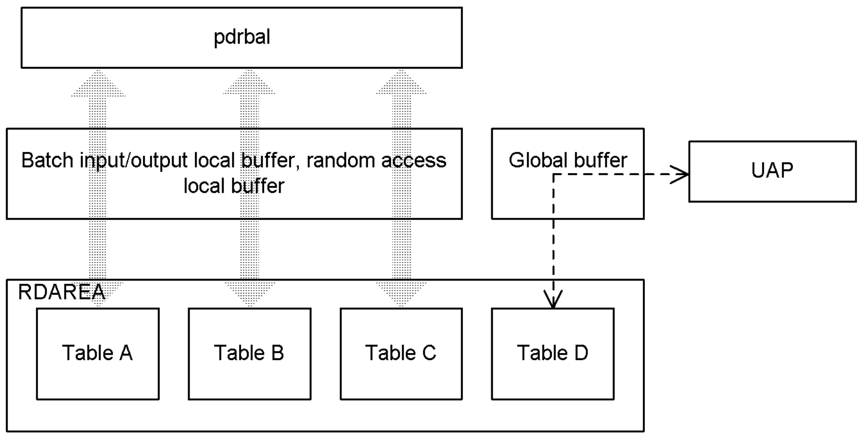
When the `-n` option is omitted, the utility uses the global buffer. In such a case, the transaction performance of a UAP that uses the global buffer drops because a large amount of global buffer space is used during rebalancing. When the `-n` option is specified, such buffer contention is eliminated. Figure 10-5 shows the relationship between `pdrbal` and buffers.

Figure 10-5: Relationship between pdrbal and buffers

- When using only a global buffer



- When using both local and global buffers



Explanation:

When only the global buffer is used (`-n` option is omitted), buffer contention occurs between `pdrbal` and the UAP.

When both local and global buffers are used (`-n` option is specified), no buffer contention occurs between `pdrbal` and the UAP. However, during rebalancing of a table with LOB columns, the utility uses the global buffer even if the `-n` option is specified.

Rules

1. When this option is omitted, the utility assumes a value of 1 and uses the global buffer. Therefore, batch I/O operations do not take place.
2. If this option is specified for a rebalancing table partitioned by the FIX hash partitioning method, the system allocates a buffer with a size of pages specified for each hash group; therefore, more memory is used than for a rebalancing table partitioned by the flexible hash partitioning method. A hash group is one of the 1024 groups created by HiRDB based on the result of hashing the partitioning key. The utility allocates RDAREA segments to each of these groups and stores data during the rebalance operation.
3. When this option is omitted (in which case the global buffer is used), at least the following number of buffer sectors is needed to achieve reasonable performance:
```
Number of buffer sectors per RDAREA
   = 1024 ÷ number of RDAREAs containing rebalancing table
     × 2 + 3
```

(8) -m *progress-message-output-interval*

~ <unsigned integer> ((1-1000)) <<10>>

Specifies in units of 10,000 lines an interval at which a message is displayed indicating the progress of the current process.

Progress messages are output for each RDAREA.

(9) *control-information-filename*

~ <pathname>

Specifies the name of the control information file that contains the control statements of `pdrbal`.

Table 10-4 lists the control statements that can be specified in the control information file. For details about each control statement, see Sections *10.3.3 index statement (specification of index information file information)* to *10.3.10*.

Table 10-4: Control statements specifiable in the control information file

Control statement (description)	Operation mode (-k option)	
	Shared mode (share)	Exclusive mode (exclusive)
`index` statement (Specification of index information file information)	—	Y
`idxwork` statement (Specification of index information file directory)	—	Y

Control statement (description)	Operation mode (-k option)	
	Shared mode (share)	Exclusive mode (exclusive)
`sort` statement (Specification of sort work directory information)	—	Y
`execstop` statement (Specification of `pdrbal` execution time information)	Y	Y
`unld_func` statement (Specification of constructor parameter reverse creation function)	Y*	Y*
`reld_func` statement (Specification of constructor function)	Y*	Y*
`report` statement (Specification of execution result file)	Y	Y
`option` statement (Specification of optional functions)	Y	Y

Y: Specifiable.

— : Not specifiable.

* With some plug-ins, specification is required. For details, see the applicable plug-in documentation.

The following rules apply to the files and directories that are specified in the control statements:

1. You must grant access privileges to the HiRDB administrator in advance. When some control statements or operands are omitted, the system assumes that the applicable file is to be created in the `/tmp` or `/usr/tmp` directory; therefore, you must also grant access privileges to the `/tmp` or `/usr/tmp` directory.

10.3.3 index statement (specification of index information file information)

When creating an index in the batch index creation mode (`-i c`), the `index` statement specifies information about an index information file to which index information is output.

Criteria

If possible, always specify the `index` statement in order to avoid a shortage of space in the `/tmp` directory.

If there are many indexes or index storage RDAREAs, you should specify the `idxwork` statement.

Rules

1. Specify one `index` statement for each index storage RDAREA added by `ALTER TABLE`.
2. There is no need to specify an `index` statement for a table with no index defined.
3. If the `index` statement is omitted and the `idxwork` statement is also omitted, the utility creates an index information file in the `/tmp` directory.
4. If both `index` and `idxwork` statements are specified, the `index` statement takes effect.
5. If a specified index information file already exists, the utility overwrites the file.
6. When the index creation process terminates normally, the system automatically deletes the specified index information file.

(1) Format

```
index index-identifier[RDAREA-name] index-information-filename
```

(2) Explanation

(a) index-identifier

~ <identifier> ((1-30))

Specify the identifier of the index.

The system treats an index identifier enclosed in double quotation marks (") as case sensitive; otherwise, the system treats it as all uppercase letters. Enclose an index identifier in double quotation marks if it contains a space.

(b) RDAREA-name

~ <identifier> ((1-30))

Specify the name of the index storage RDAREA that was added by `ALTER TABLE`.

The system treats an RDAREA name enclosed in double quotation marks (") as case sensitive; otherwise, the system treats it as all uppercase letters. Enclose an RDAREA name in double quotation marks if it contains a space.

(c) index-information-filename

~ <pathname>

Specify the absolute path name of the index information file to which index information is to be output.

This must be the name of a regular file.

10.3.4 idxwork statement (specification of directory for index information files)

The `idxwork` statement specifies the name of a directory in which index information files are to be created automatically when the `index` statement is omitted in the batch index creation mode (`-i c`).

Criterion

If possible, always specify the `idxwork` statement in order to avoid a shortage of space in the `/tmp` directory.

Rules

1. If the `index` and `idxwork` statements are both omitted, the utility creates index information files in the `/tmp` directory on the server that contains the index storage RDAREA added by `ALTER TABLE`.
2. For a HiRDB/Single Server, specify only one `idxwork` statement.

   For a HiRDB/Parallel Server, specify as many `idxwork` statements as there are servers that contain the index storage RDAREAs added by `ALTER TABLE`.
3. If both `idxwork` and `index` statements are specified, the `index` statement takes effect, in which case the `idxwork` statement is ignored.
4. When the index creation process terminates normally, the system automatically deletes the index information files from the specified directory.

(1) Format

```
idxwork [server-name] directory-name
```

(2) Explanation

(a) server-name

～ <identifier> ((1-8))

Specify the name of the server used to create the directory for index information files. For a HiRDB/Single Server, do not specify the server name.

(b) directory-name

~ <pathname> ((1-255))

Specify the absolute pathname of the directory in which index information files are to be created.

(3) Names of the index information files to be created

The utility assigns the following name to each created index information file:

directory-name/
`INDEX`-*index-name*-*index-storage-RDAREA-name*-*unique-character-string*

The following shows an example:

- Conditions

  Index name: `IDX1`

  Index storage RDAREA: `USER01`

  Contents of `idxwork` statement: `idxwork /hd0400`

- Name of the index information file to be created

  `/hd0400/INDEX-IDX1-USER01-aaaa00001`

10.3.5 sort statement (specification of work directory for sorting)

The `sort` statement specifies information about a work file for sorting that is used when an index is created in the batch index creation mode (`-i c`).

Criterion

If possible, always specify the `sort` statement in order to avoid a shortage of space in the `/tmp` directory.

Rules

1. If the `sort` statement is omitted, the utility assumes the `/tmp` directory on the server that contains the index storage RDAREAs.
2. For a HiRDB/Single Server, specify only one `sort` statement.

   For a HiRDB/Parallel Server, specify as many `sort` statements as there are servers that contain the index storage RDAREAs added by `ALTER TABLE`.

(1) Format

```
sort [server-name] directory-name[,buffer-size-for-sorting]
```

(2) Explanation

(a) server-name

~ <identifier> ((1-8))

Specify the name of the server used to create the work file for sorting.

For a HiRDB/Single Server, do not specify the server name.

(b) directory-name

~ <pathname> ((1-255))

Specify the absolute pathname of the directory in which the sort work file is to be created.

(c) buffer-size-for-sorting

~ <unsigned integer> ((128-2097152)) <<1024>>

Specify, in KB, the size of memory that is to be used as the buffer.

The system allocates this buffer at the single server for a HiRDB/Single Server and at the back-end server for a HiRDB/Parallel Server.

The sort process creates a temporary work file in a specified directory. You can use the following formula to determine a buffer size that minimizes the file size. This is just a guideline. If there is not enough memory, avoid using a large value.

• In 32-bit mode HiRDB

$$\text{Buffer size (bytes)} \geqq \frac{R+7}{2} + \sqrt{(B+8) \times n \times A + \frac{(R+7)^2}{4}} + C$$

• In 64-bit mode HiRDB

$$\text{Buffer size (bytes)} \geqq \frac{R+15}{2} + \sqrt{(B+8) \times n \times A + \frac{(R+15)^2}{4}} + C$$

n: Number of data items to be rebalanced. For a repetition column, the number of data items means the number of elements, not the number of rows.

k: Key length (calculated as a maximum value). For details about how to determine the key length, see the example of calculating the number of index storage pages in the *HiRDB Version 8 Installation and Design Guide.*

x: `10` if all key component columns have a fixed length; `12` if at least one of the key component columns has a variable length.

c:

Number of columns composing the index.

y:

For Linux version; 2; otherwise, 1.

z:

For a variable-length multicolumn index, *c* x 4; otherwise, 0.

K:

For a variable-length multicolumn index, *k* + *c* + 8; otherwise, *k* + 12.

N:

For a variable-length multicolumn index, (*c* x 2) + *y*; otherwise, 3 + *y*.

R:

k + x + z

A:

For 32-bit mode HiRDB, *R* + (*K* + 8) + 28; for 64-bit mode HiRDB, *R* + (*K* + 8) + 56.

B:

For 32-bit mode HiRDB, *R* + (*K* + 8) + 56; for 64-bit mode HiRDB, *R* + (*K* + 8) + 104.

C:

For 32-bit mode HiRDB, 2092 + (*N* x 32) + (*K* + 8); for 64-bit mode HiRDB, 2112 + (*N* x 32) + (*K* + 8).

(3) Note

Do not allocate NFS to the directory specified in the `sort` statement. If NFS is allocated, twice as much space is required for the work file for sorting as when a local file is used. Also, problems occur related to retention of work files for sorting.

10.3.6 execstop statement (specification of pdrbal execution time)

The `execstop` statement specifies a `pdrbal` execution time to stop `pdrbal` at a specific time.

Criterion

Specify the `execstop` statement to avoid executing `pdrbal` when there are many online transactions, such as during business hours.

Rules

1. When the `execstop` statement is omitted, the utility continues the rebalance

operation until it is completed.

2. Even when the specified termination time is reached, the utility continues processing until all row data (including indexes) involved in the rebalance operation has been moved. Therefore, the utility's processing may not terminate by the specified time. In the batch index creation mode, the index is created in the batch mode after data has been moved. Take this into account when you specify the termination time.

(1) Format

```
execstop time,pdrbal-execution-time
```

(2) Explanation

(a) pdrbal-execution-time

Specify the `pdrbal` execution time (*HH*:*MM*).

HH: Hours (00-168)

MM: Minutes (00 or 30)

This value cannot be greater than 168 hours (one week). If `00` is specified for both *HH* and *MM*, the utility terminates immediately without executing the rebalance operation (however, at the startup, `pdrbal` executes the access procedure for search, deletion, and update processing, deletion of SQL objects, and recompilation of a procedure/ function for which an SQL object is invalid).

Example:

Execute `pdrbal` from 21:00 to 9:00 (12 hours) when there is not much traffic:
`execstop time,12:00`

10.3.7 unld_func statement (specification of constructor parameter reverse creation function)

When `pdrbal` is executed on a table with columns of abstract data type, the utility uses the plug-in's unload facility to rebalance the table. The `unld_func` statement specifies a constructor parameter reverse creation function that reverses the creation of data values for the abstract data type during the unload operation.

Criterion

Specify the `unld_func` statement to rebalance a table with an abstract data type provided by a plug-in that has an unloading facility.

Rules

1. Be sure to specify the `unld_func` statement if the table has an abstract data

type provided by a plug-in and the plug-in has an unload facility.

2. You can specify as many `unld_func` statements as there are abstract data types defined for the table columns.
3. When specifying an `unld_func` statement, also specify a `reld_func` statement as a pair.

(1) Format

```
unld_func type=[authorization-identifier.]abstract-data-type-name,
       func=function-name (argument-type[,argument-type...])
       [,func=function-name (argument-type[,argument-type...])...]
```

(2) Explanation

(a) type=[authorization-identifier.]abstract-data-type-name

Specify the authorization identifier and name of the abstract data type.

Rules

1. If the authorization identifier is omitted, the authorization identifier of the user who defined the abstract data type (normally `MASTER`) is assumed.
2. If the authorization identifier or abstract data type name contains a lowercase letter or a space, enclose it in double quotation marks (`"`).

(b) func=function-name (argument-type[,argument-type...])

Specify the name and argument type of the constructor parameter reverse creation function. For details about the name and argument type of a constructor parameter reverse creation function, see the applicable plug-in documentation.

function-name

Specify the name of the constructor parameter reverse creation function.

argument-type

Specify the data type of the argument of the constructor parameter reverse creation function. Table 10-5 shows the data types of the argument.

Table 10-5: Data types of argument (pdrbal)

Specification method	Remarks
`integer`	`INT32`
`smallint`	`INT16`

`char`
`nchar`
`mchar`
	`CHAR8`

Specification method	Remarks

`varchar`
`nvarchar`
`mvarchar`
	`p_pdb_varchar_t`

`float`	`DOUBLE64`
`smallflt`	`REAL64`
`blob`	`BLOB`
abstract-data-type-name	None
binary	p_pdb_binary_t

10.3.8 reld_func statement (specification of constructor function)

When `pdrbal` is executed on a table with columns of abstract data type, the utility uses the plug-in's unload facility to rebalance the table. The `reld_func` statement specifies a constructor function that generates values of an abstract data type during the reload operation.

Criterion

Specify the `reld_func` statement to reorganize a table with an abstract data type provided by a plug-in that has an unloading facility.

Rules

1. Be sure to specify the `reld_func` statement if the table has an abstract data type provided by a plug-in and the plug-in has an unload facility.
2. You can specify as many `reld_func` statements as there are abstract data types defined for table columns.
3. When specifying a `reld_func` statement, also specify an `unld_func` statement as a pair.

(1) Format

```
reld_func type=[authorization-identifier.]abstract-data-type-name,
       func=function-name  (argument-type[,argument-type...])
```

(2) Explanation

(a) type=[authorization-identifier.]abstract-data-type-name

Specify the authorization identifier and name of the abstract data type.

Rules

1. If the authorization identifier is omitted, the authorization identifier of the user who defined the abstract data type (normally `MASTER`) is assumed.
2. If the authorization identifier or abstract data type name contains a lowercase letter or a space, enclose it in double quotation marks (`"`).

(b) func=function-name (argument-type[,argument-type...])

Specify the name and argument type of the constructor function. For details about the name and argument type of a constructor function, see the applicable plug-in documentation.

function-name

Specify the name of the constructor function.

argument-type

Specify the data type of the argument of the constructor function. For details about the format of the argument data type, see Section *10.3.7 unld_func statement (specification of constructor parameter reverse creation function)*. You cannot specify the name of an abstract data type.

10.3.9 report statement (specification of a process results file)

The `report` statement specifies a file to which the processing result of `pdrbal` is to be output.

Criterion

Specify the `report` statement to create a process results file in a directory other than `/tmp` on the host where `pdrbal` is executed.

Rules

1. When the `report` statement is omitted, the utility creates a process results file in `/tmp` on the host where `pdrbal` is executed.
2. The contents of the process results file are not guaranteed until `pdrbal` is terminated.
3. If `pdrbal` terminates with return code = `8`, the utility may not output the part of the information following `Moved data information`. If output, the information was obtained after rollback (at the commit point immediately before the occurrence of the error).

(1) Format

```
report process-results-filename
```

(2) Explanation

(a) process-results-filename

~ <pathname>

Specify the absolute path name of the file to which the processing result is to be output.

This must be the name of a regular file located at the host where `pdrbal` is executed.

(3) Name of the process results file created when the report statement is omitted

The utility assigns the following name to a created process results file:
`/tmp/REPORT-`*table-identifier-unique-character-string*

For example, if the table name is `TABLE1`, then the following name is assigned to a created process results file:

```
/tmp/PEPORT-TABLE1-aaaa00001
```

(4) Output format

The following shows the output format of a `pdrbal` process results file:

```
pdrbal VV-RR(Object Option) *** DB REBALANCE *** yyyy-mm-dd
HH:MM:SS [1]
--------------------------------------------------------------
----------

*** Table rebalance processing list ***

 Schema name  : robinson  [2]
 Table name   : rev_table  [3]
 Table status : completed  [4]

 ** Moved data information **
 [5]  [6]         [7]                       [8]          [9]
 No. Server   Table rdarea name           Remove row    Insert
row
  1 sds     user01                     1,000,000          0
  2 sds     user02                      2,000,000          0
  3 sds     user03                             0     3,000,000
                                Total   3,000,000     3,000,000
                                             [10]          [11]
 ** Rebalance information **
 [12] [13]         [14]                          [15]
 No. Server   Table rdarea name                  Status
   1 sds      user01                             finished
   2 sds      user02                             finished
   3 sds      user03

 *** Exclusive execute information ***
```

```
  Start time : 2000-05-05 11:12:13  [16]
  End   time : 2000-05-05 19:20:21  [17]

 pdrbal terminated, return code=0  [18]
```

Explanation

1. Header information for the process results

   `VV-RR`: Version number

   `yyyy-mm-dd HH:MM:SS`: Date and time `pdrbal` was started
2. Schema name
3. Table identifier
4. Table rebalancing status

   `completed`: Rebalancing completed

   `processing`: Rebalancing underway
5. Sequence number of the source or target RDAREA
6. Name of the server containing the source or target RDAREA
7. Name of the source or target RDAREA
8. Number of rows moved (deleted) from the source RDAREA
9. Number of rows moved (added) to the target RDAREA
10. Total number of rows moved (deleted) from the source RDAREAs
11. Total number of rows moved (added) to the target RDAREAs
12. Sequence number of table storage RDAREA
13. Name of the server containing the table storage RDAREA
14. Number of table storage RDAREA
15. RDAREA rebalancing status

    `finished`: Rebalancing completed

    `processing`: Rebalancing underway

    `error`: Terminated with error

    blank: Added RDAREA
16. Date and time rebalance operation started
17. Date and time rebalance operation terminated

18. `pdrbal`'s return code

10.3.10 option statement (specification of optional functions)

The `option` statement specifies `pdrbal`'s optional functions.

Criteria

Specify the `option` statement to change the interval at which the utility is to check whether or not `pdrbal` can be executed in the shared mode.

Rule

1. You can specify only one `option` statement.

(1) Format

```
..option [check_itv=executability-checking-interval]
```

(2) Explanation

(a) check_itv=executability-checking-interval

~ <unsigned integer> ((10-180000)) <<10>>

In the shared mode (`-k share`), `pdrbal` checks whether or not there is any UAP that accesses the target table. If there is no such UAP, `pdrbal` moves the data. Specify this executability checking interval in milliseconds.

In the exclusive mode (`-k exclusive`), you cannot specify this option.

Criteria

Normally, the default value is used.

If the `pdrbal` workload is large and online applications may be affected adversely, specifying a value that is greater than the default value increases the monitoring interval at which `pdrbal` checks UAP operation, which is to the benefit of the online applications.

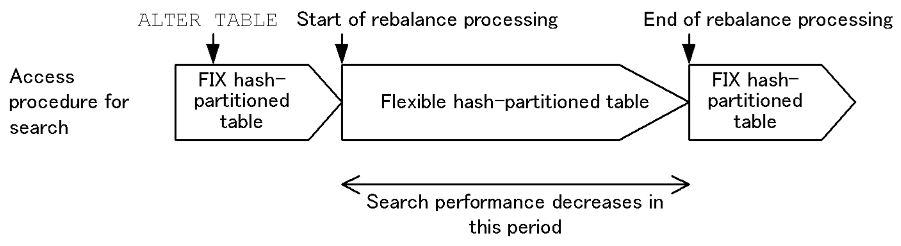
10.4 Notes about the execution of SQL on FIX hash-partitioned tables during a rebalance operation

For a table partitioned by the FIX hash partitioning method, you can specify an RDAREA in which data is to be stored or searched for by the hash key value. However, if you add an RDAREA to a table partitioned by the FIX hash partitioning method (`HASHA` to `HASHF`), the utility assumes the access procedure for a table partitioned by the flexible hash partitioning method during search operation. If this happens, the search performance decreases for the following SQL:

- Search operation specifies a partitioning key as the condition.
- The column specified in the `GROUP BY` clause contains all partitioning keys.

Figure 10-6 shows the relationship between the rebalance processing and the period during which search performance decreases. While rebalance processing is underway, the number of processes required for SQL execution increases.

Figure 10-6: Relationship between rebalance processing and the period during which search performance decreases


ALTER TABLE
Start of rebalance processing
End of rebalance processing
Access procedure for search
FIX hash-partitioned table
Flexible hash-partitioned table
FIX hash-partitioned table
Search performance decreases in this period

For a table partitioned by the flexible hash partitioning, the rebalance processing has no effect on the access procedure for a search operation.

10.5 Whether or not SQL is executable during a rebalance operation

Table 10-6 shows whether or not SQL is executable during a rebalance operation.

Table 10-6: Whether or not SQL is executable during a rebalance operation

SQL statement	**Rebalance operation**		**While rebalance operation is underway or stopped**
	Shared mode	**Exclusive mode**	
`SELECT` statement	Y	—	Y
`UPDATE` statement	Y[1]	—	Y[1]
`INSERT` statement	Y[2]	—	Y[2]
`DELETE` statement	Y	—	Y
`LOCK` statement	Y	—	Y
`PURGE TABLE` statement[3]	—[4]	—	Y
`DROP TABLE`	—[4]	—	Y
`CREATE INDEX`	—[4]	—	Y
`DROP INDEX`	—[4]	—	Y
`ALTER TABLE`	—[4]	—	—

Y: Executable.

—: Not executable.

[1] A column constituting a unique key index cannot be updated (updating such a column results in an error).

[2] The SQL statement is not executable if a unique key index is defined for the table that is to be rebalanced (executing the SQL statement results in an error).

[3] If you add an RDAREA and then execute the `PURGE TABLE` statement or reinitialize the RDAREA before executing `pdrbal`, storage processing does not take place on the added RDAREA because rebalance initialization processing has not been executed. If you execute the `PURGE TABLE` statement or reinitialize the RDAREA before `pdrbal` terminates with return code `0`, SQL search performance on a FIX hash-partitioned table remains low because rebalance completion processing has not been executed. In this case, execute `pdrbal` immediately after executing the `PURGE TABLE` statement or reinitializing the RDAREA. This results in the execution of rebalancing

initialization or completion processing.

4 If a recovery-unnecessary front-end server is used, a definition SQL or `PURGE TABLE` statement may be executed during execution of `pdrbal` (if a recovery-unnecessary front-end server is not used, the front-end server prevents a definition SQL or `PURGE TABLE` statement from being executed illegally by allocating the locked resources). Therefore, make sure that neither a definition SQL nor a `PURGE TABLE` statement is executed during execution of `pdrbal`. If you execute `pdrbal` in the exclusive mode, a definition SQL statement is placed on lock-release wait because the table is locked by the back-end server.

10.6 Notes

1. Following are `pdrbal`'s return codes:

   `0`: Normal termination (rebalancing completed)

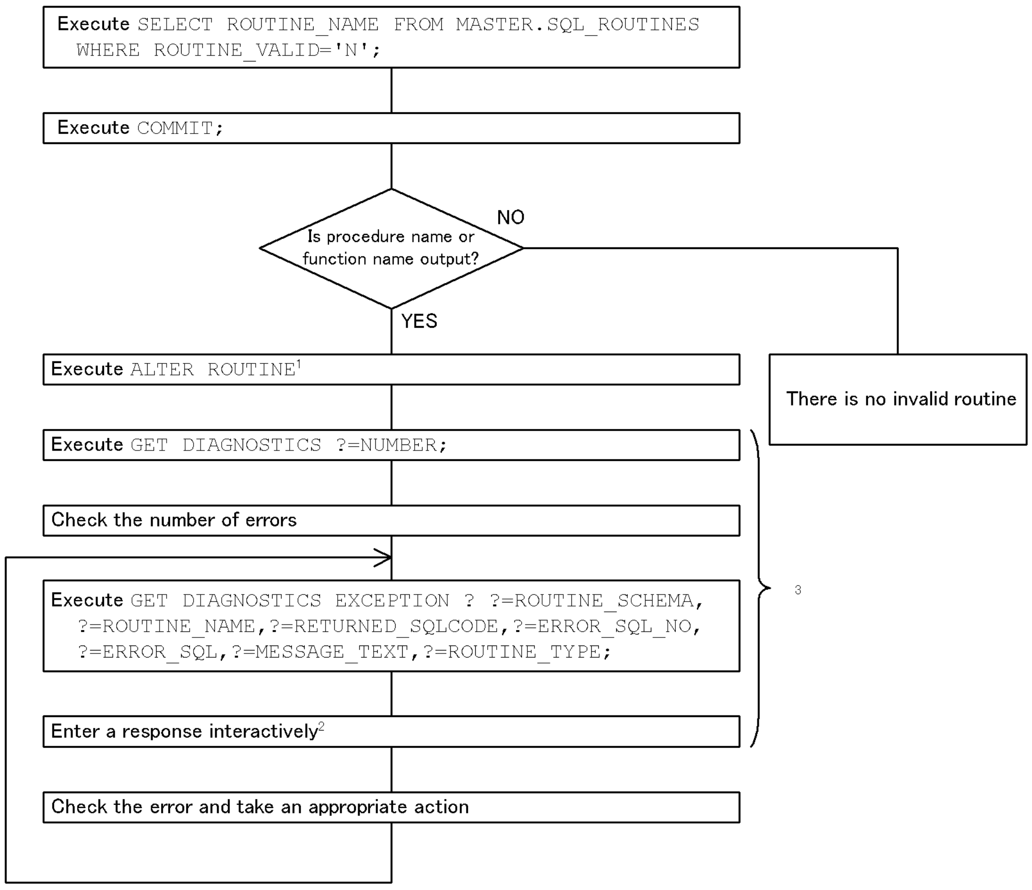
   `4`: Normal termination (rebalancing underway)

   `8`: Abnormal termination
2. You cannot execute a definition SQL statement on a table being rebalanced.
3. No other UAP or utility can access a table or index that is being rebalanced in the exclusive mode (`-k exclusive`).
4. If a list has been created based on a rebalancing table, a search process using this list may not produce a valid result during a rebalance operation. In this case, re-create the list before using it for a search process.
5. When a rebalance operation is underway in the shared mode, if you specify a commit between acquisition of row identifier and actual data manipulation to search, update, or delete data using the row identifier with a UAP, the result may be invalid.
6. You cannot execute `pdrbal` if the rebalancing table contains a number of table storage RDAREAs, that does not match the number of index storage RDAREAs (non-partitioning key index that is not partitioned within the same server). To rebalance such a table, delete the non-partitioning key index that is not partitioned within the same server, then execute `pdrbal`. After the rebalance operation is completed, redefine the index. You cannot define such an index during the rebalance operation. If you define a non-partitioning key index for a rebalancing table, store the index in the same number of index storage RDAREAs as the table storage RDAREAs within the same server.
7. The database update log information output by `pdrbal` is not applicable to HiRDB DataReplicator's data linkage facility.
8. For index deletion or creation for the data being rebalanced, the utility always uses the global buffer regardless of the specification of the `-n` option. Therefore, execute `pdrbal` in an environment where sufficient global buffer is available. Especially for the source RDAREA, you should allocate a global buffer for each index. If there is not enough global buffer, contention may occur on the buffer, resulting in the I/O busy status, thereby reducing the performance.
9. If there are multiple generations of RDAREAs while the inner replica facility is being used, `pdrbal` terminates with an error.
10. An attempt to rebalance a table that contains a user LOB RDAREA that is in frozen update status will cause `pdrbal` to terminate with an error.

11. During rebalancing initialization and completion processing, `pdrbal` recompiles all procedures and functions in the database for which an SQL object is invalid. An error results if any such procedures or functions cannot be recompiled. If an error occurs, check the error message, take an appropriate action, then recompile the procedure or function using `ALTER PROCEDURE` or `ALTER ROUTINE` (recompilation does not take place during re-execution of `pdrbal`).

    To check for a procedure or function with an invalid SQL object that cannot be recompiled prior to the execution of `pdrbal`, use SQL Executer. Figure 10-7 shows the procedure for checking for a procedure or function with an invalid SQL object that cannot be recompiled.

Figure 10-7: Procedure for checking for a procedure or function with invalid SQL object that cannot be recompiled


Execute SELECT ROUTINE_NAME FROM MASTER.SQL_ROUTINES WHERE ROUTINE_VALID='N';
Execute COMMIT;
Is procedure name or function name output?
NO
YES
Execute ALTER ROUTINE[1]
There is no invalid routine
Execute GET DIAGNOSTICS ?=NUMBER;
Check the number of errors
Execute GET DIAGNOSTICS EXCEPTION ? ?=ROUTINE_SCHEMA, ?=ROUTINE_NAME,?=RETURNED_SQLCODE,?=ERROR_SQL_NO, ?=ERROR_SQL,?=MESSAGE_TEXT,?=ROUTINE_TYPE;
Enter a response interactively[2]
Check the error and take an appropriate action
3

Note: You need the DBA privilege to complete the procedure beginning at the step that involves execution of `ALTER ROUTINE;`.

[1] `ALTER ROUTINE` executed at this step results in an error.

[2] Enter responses sequentially from 1 through the number of errors.

[3] To execute a GUI version of SQL Executer, execute `GET DIAGNOSTICS;`.

12. Table 10-7 shows the availability of file medium and large files for `pdrbal`.

Table 10-7: Availability of file medium and large files for pdrbal

File	Medium		Large file
	Regular file	Fixed- or variable-length block tape	
Control information file	Y	—	Y
Index information file	Y	—	Y
Work file for sorting	Y	—	Y
Process results file	Y	—	Y

Y: Usable.

— Not usable.

13. You cannot execute `pdrbal` if a falsification prevented table is to be rebalanced and that falsification prevented table is in reload-not-completed data status.
14. If `pdrbal` is executed in the shared mode in a HiRDB/Parallel Server system, all front-end servers to which a recovery-unnecessary front-end server is not applied must be active.
15. `pdrbal` cannot be executed while the table is in check pending status (value of the `CHECK_PEND` or `CHECK_PEND2` column in the `SQL_TABLES` data dictionary table is `C` (check pending status)).
16. If you selected `utf-8` as the character encoding in the `pdsetup` command, you may be able to use a file with a BOM as the input file for `pdrbal`. Table 10-8 shows whether or not files with a BOM can be used with `pdrbal`. Note that even when a file with a BOM is used as the input file for `pdrbal`, the BOM is skipped. No BOM is included in the file that is output by `pdrbal`.

Table 10-8: Whether or not files with a BOM can be used in pdrbal (applicable to UTF-8)

Control statement	Input file	Use of file with a BOM
—	Control information file	Y
`index`	Index information file	N

Legend:

Y: Can be used

N: Cannot be used

— : Not applicable

10.7 Error handling procedures

1. If `pdrbal`'s return code is `8`, check the error message, correct the error, then re-execute `pdrbal`.
2. If the `KFPL33003-I` message (`STATUS=END`) is displayed during the execution of `pdrbal`, table rebalance processing has terminated normally. Therefore, once this message is issued, there is no need to re-execute `pdrbal`, even if an error occurs thereafter.
3. If an error occurs during batch index creation in the exclusive mode (return code = `8`), eliminate the cause of the error and then execute batch index creation (`-k ixmk`) using `pdrorg` along with the index information files for the index that resulted in an error. Then, execute `pdrbal`.
4. If rollback occurs during the execution of `pdrbal`, the action to be taken depends on the log acquisition method (`-l` option). Table 10-9 shows the action to be taken in the event of a rollback during the execution of `pdrbal`.

Table 10-9: Action to be taken in the event of a rollback during the execution of pdrbal

Log acquisition method (-l option)		**pdrbal processing**	
		Rebalancing	**Batch index creation (exclusive mode)**
`a`	Database status	The row data that was rolled back has not been moved. The preceding row data have already been moved.	Index is placed in unfinished status.
	Action	Re-execute `pdrbal`.	Execute batch index creation (`-k ixmk`) using `pdrorg`. In this case, use the index information files that resulted in an error.
`n`	Database status	There is no guarantee.	There is no guarantee.
	Action	Restore the database from its backup copy, then re-execute `pdrbal`.	Reinitialize the corresponding index storage RDAREAs and re-create the index (`-k ixrc`) using `pdrorg`.

Chapter

11. Free Page Release Utility (pdreclaim)

This chapter explains the free page release utility (`pdreclaim`) that enables you to release free pages (used free pages) during online operation.

11.1 Overview

11.1.1 About free pages

Before describing the `pdreclaim` command, this section describes pages and segments. Pages and segments can be in the statuses shown in Table 11-1.

Table 11-1: Page and segment status

Page and segment status	Description
Used page	A page containing table or index data. A page that is filled with data and in which no more data can be stored is called a *full page*; a page that contains no data due to data deletion is called a *used free page*.
Unused page	A page that has never been used.
Free page	A page that contains no data. Used free pages and unused pages are both free pages.
Used segment*	A segment containing table or index data. A segment that is filled with data and to which no more data can be added is called a *full segment*; a segment whose pages are all empty due to data deletion (used free pages or unused pages) is called a *used free segment*.
Unused segment	A segment that has never been used. All tables (or indexes) in RDAREAs can use unused segments.
Free segment	A segment that contains no data. Used free segments and unused segments are both free segments.

* A used segment is available only to the table or index whose data is currently stored in it. No other table or index can use such a segment.

11.1.2 Overview of pdreclaim

Continuous database operation and repeated data additions, updating, and deletions affect the arrangement of indexes and data in a database, resulting in reduced data storage efficiency and processing performance. You can repair the data and index arrangement by executing the database reorganization utility (`pdrorg` command). However, you must stop online services during execution of `pdrorg` because a table cannot be referenced or updated while it is being processed by the `pdrorg` command (except when the inner replica facility is used).

The free page release utility (`pdreclaim`) is able to release used free pages, which is an element of the reorganization processing executed by the `pdrorg` command, during online operation. The `pdreclaim` command executes `pdrorg` internally. Note that used free pages cannot be released when an application program that places the table in the lock mode (EX) is executing or when a utility is executing. Free segments can be released only when neither an application program nor a utility is accessing the subject RDAREA.

Use `pdreclaim` in the following cases:

- Database reorganization is not appropriate to the system's mode of operation because it is not feasible to stop online service.*
- The database reorganization interval (operation cycle) needs to be increased without compromising storage efficiency.

* For details about the limitations when database reorganization is not performed by `pdreclaim`, see *11.1.4 Differences in functions from pdrorg*. If you cannot use `pdreclaim` due to the limitations, you must use the online reorganization function that combines `pdrorg` and the inner replica facility.

11.1.3 Functions of pdreclaim

(1) Releasing used free pages

`pdreclaim` has the following two functions:

- Releasing used free pages from a table
- Releasing used free pages from a table's indexes

Executing `pdreclaim` without the `-j` option specified releases used free pages. If used free segments are created as a result, executing `pdreclaim` with the `-j` option specified releases the used free segments. The following table shows whether or not `pdreclaim` can be executed depending on the specification of the `-j` option:

Option specification in pdreclaim	Releases used free pages	Releases used free segments
`-j` option omitted	Y	N
`-j` option specified	N	Y

Legend:

Y: Executed

N: Not executed

Note that you cannot release used free segments without first releasing used free pages. However, if either of the following operations is executed, used free segments are created; therefore, `pdreclaim` with the `-j` option specified can release used free segments without releasing the used free pages by `pdreclaim` without the `-j` option specified.

- Deletion of data from a locked table

- Execution of `pdload` (with `yes` specified in the `nowait` operand of the `option` statement) during data loading that results in abnormal termination and rollback

■ Effects of releasing used free pages from a table

Releasing used free pages from a table provides the following advantages:

- Improvement of global search performance

  Pages whose usage percentage is 0% become unused pages that no longer need to be searched during a global search, resulting in an improvement of global search performance.

- Elimination of errors during execution of `UPDATE` and `INSERT` statements on branch rows

  When there are no more unused pages, an `UPDATE` or `INSERT` statement on branch rows results in an error (`KFPA11756-E`). You can eliminate such errors by changing the pages whose usage percentage is 0% to unused pages.

- Elimination of page compaction during execution of `INSERT` and `UPDATE` statements

  Because page compaction is executed on all used free pages in the batch mode, you can eliminate the page compaction that is executed as an extension of `INSERT` and `UPDATE` statements.

■ Effects of releasing used free pages from indexes

Releasing used free pages from indexes provides the following advantages:

- Elimination of area shortages when used free pages are available
- Elimination of adverse effects on performance that are caused by referencing used free pages during index searches

After releasing used free pages from indexes, the utility first allocates those unused pages that were released by means of used free space search processing. The utility allocates unused pages in used segments preferentially, thereby achieving efficient space utilization and reducing the overhead of used free space search processing.

(a) Releasing used free pages in a table

Figure 11-1 provides an overview of releasing used free pages in a table.

Figure 11-1: Overview of releasing used free pages in a table

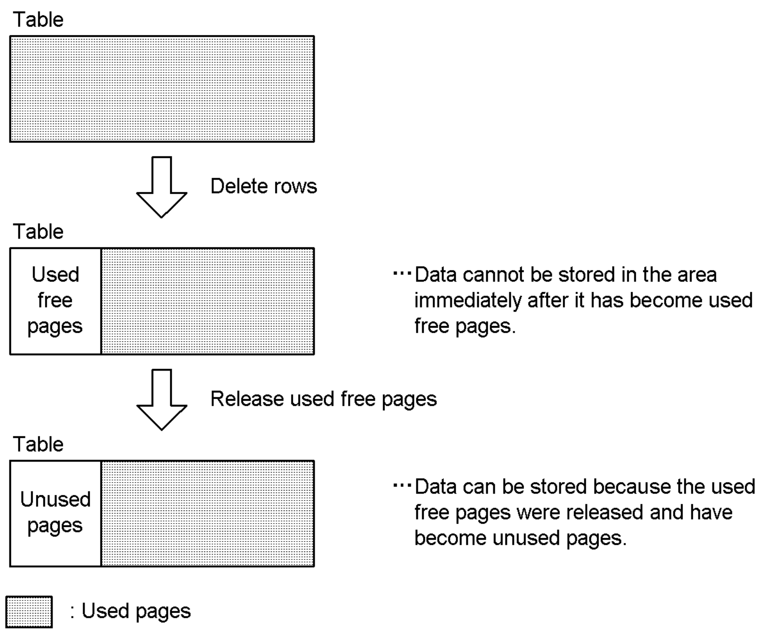


Table
Delete rows
Table
Used free pages
...Data cannot be stored in the area immediately after it has become used free pages.
Release used free pages
Table
Unused pages
...Data can be stored because the used free pages were released and have become unused pages.
: Used pages

By releasing used free pages in a table, you can select release of used free segments, release of used free pages, or execution of page compaction. Note that release of used free pages in a table is not applicable to LOB columns or columns of an abstract data type with the LOB attribute.

- Releasing used free segments and used free pages

Deletion of a large amount of data results in used free pages that contain no data but which remain allocated. For all tables other than `SEGMENT REUSE` tables (tables for which the `SEGMENT REUSE` option is specified in `CREATE TABLE` or `ALTER TABLE`), HiRDB uses unused pages as well as unused segments for storing data. If no new space is available, HiRDB reuses used free pages, in which case performance is not as good as when new space can be allocated due to the overhead of searching for free space in the used pages. Even with a `SEGMENT REUSE` table, if there are used free pages, a global search involves unneeded searching of empty pages. Note that only the tables and indexes in the corresponding used segments are reusable.

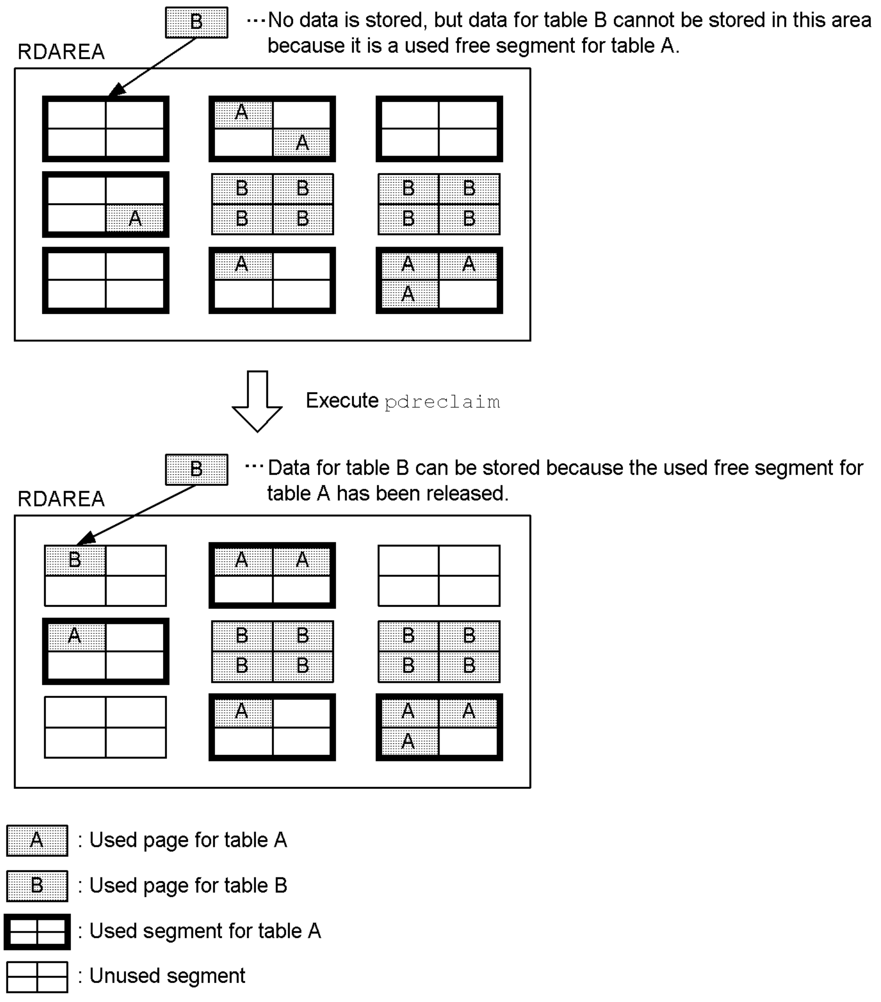
Execution of `pdreclaim` has the following effects because this utility releases used free pages that are not immediately reused:

1. A segment from which all data has been deleted becomes an unused segment; therefore, it becomes available to all tables and indexes.
2. A page from which all data has been deleted becomes an unused page; therefore,

it can be reused by the table in the corresponding used segment. Especially with a `SEGMENT REUSE` table, the effects are great because the overhead of searching the pages to be reused is eliminated.

As a part of effect 1, if the same RDAREA stores multiple tables, it is possible to prevent a shortage of RDAREA space that may result if table B cannot allocate a new unused segment because Table A has used free segments that contain no data. Figure 11-2 shows the effects of `pdreclaim` when the same RDAREA stores multiple tables.

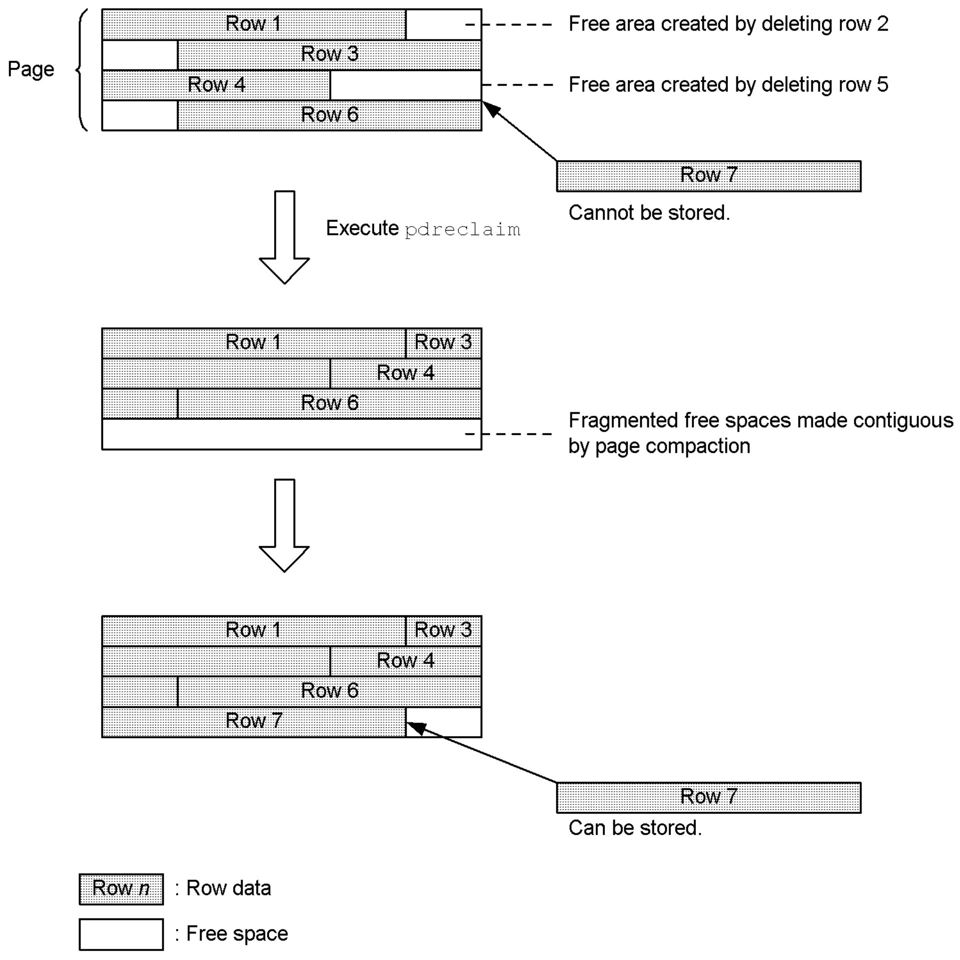
Figure 11-2: Effects of pdreclaim when the same RDAREA stores multiple tables


B
···No data is stored, but data for table B cannot be stored in this area because it is a used free segment for table A.
RDAREA
A
B
Execute pdreclaim
···Data for table B can be stored because the used free segment for table A has been released.
: Used page for table A
: Used page for table B
: Used segment for table A
: Unused segment

- Page compaction in tables

Figure 11-3 provides an overview of page compaction in tables.

Figure 11-3: Overview of page compaction in tables


Page
Row 1
Row 3
Row 4
Row 6
Free area created by deleting row 2
Free area created by deleting row 5
Row 7
Cannot be stored.
Execute pdreclaim
Fragmented free spaces made contiguous by page compaction
Row 7
Can be stored.
Row n : Row data
: Free space

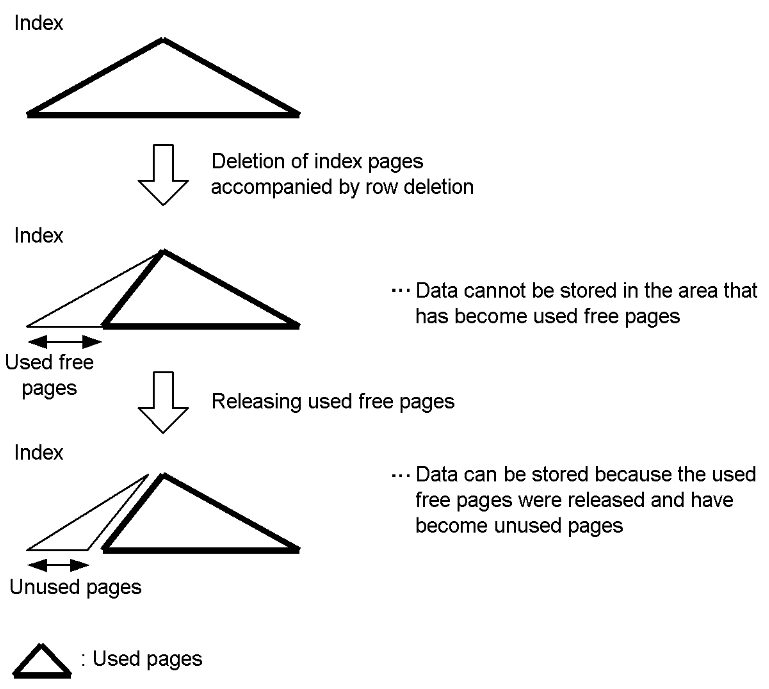
Explanation

Before execution of `pdreclaim`, any data larger than row 2 or 5 cannot be stored as is in the free space created by deleting rows 2 and 5. If these two free spaces are made contiguous, then data can be stored. Reorganizing these scattered free spaces into a single contiguous space is called *page compaction*. Page compaction is also executed as an extension of SQL statements. However, online performance is reduced when page compaction is executed as an extension of SQL statements. You can avoid such adverse effects on online performance by using `pdreclaim` to execute page compaction in advance.

(b) Releasing used free pages from indexes

Figure 11-4 provides an overview of releasing used free pages in indexes.

Figure 11-4: Overview of releasing used free pages in indexes


Index
Deletion of index pages accompanied by row deletion
Index
··· Data cannot be stored in the area that has become used free pages
Used free pages
Releasing used free pages
Index
··· Data can be stored because the used free pages were released and have become unused pages
Unused pages
: Used pages

By releasing used free pages in indexes, you can select release of used free segments or release of used free pages. Note that release of used free pages is not applicable to plug-in indexes.

- Releasing used free segments and used free pages

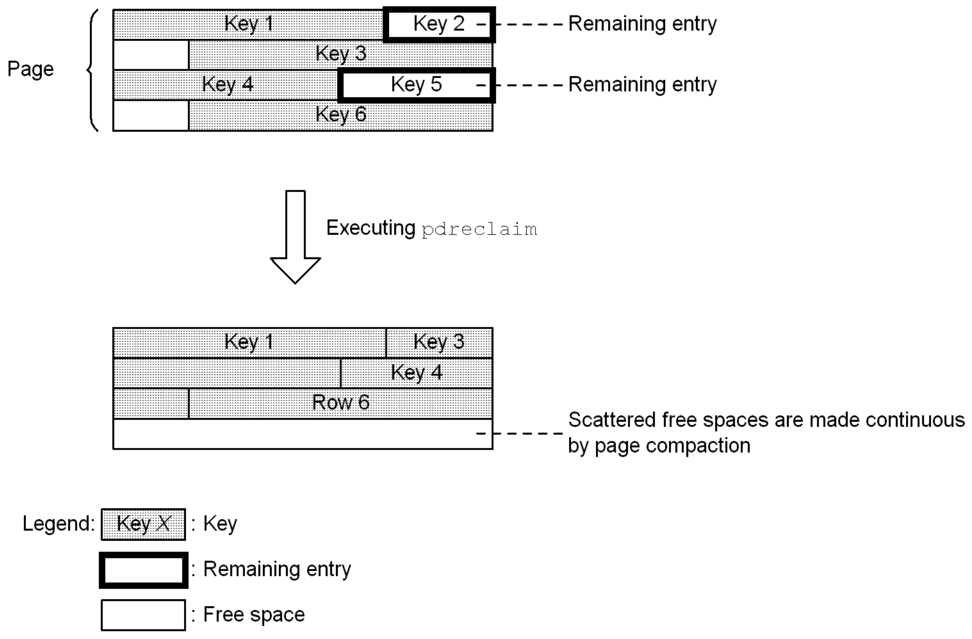
As is the case with a table, if a large amount of data is deleted from an index, unused free pages are created that are not reusable. Especially, used free pages that are created by deleting existing data with small key values remain allocated (remain connected by pointers) to make them available should the same key values be stored again; these used free pages cannot be reused when new data with a large key value is added. Therefore, in the operation mode where key values of stored data simply increase as previous data is deleted, a steadily increasing number of nonreusable used free pages is created. If you use `pdreclaim` on such an index, data with the large key values can be stored because the used free pages are released and unused pages are created. This utility can also create unused pages from intermediate pages with no downward page pointers.

- Page compaction in indexes

In indexes, page compaction is executed only when the target of free page release is a unique index.

Figure 11-5 provides an overview of page compaction in indexes.

Figure 11-5: Overview of page compaction in indexes



Explanation

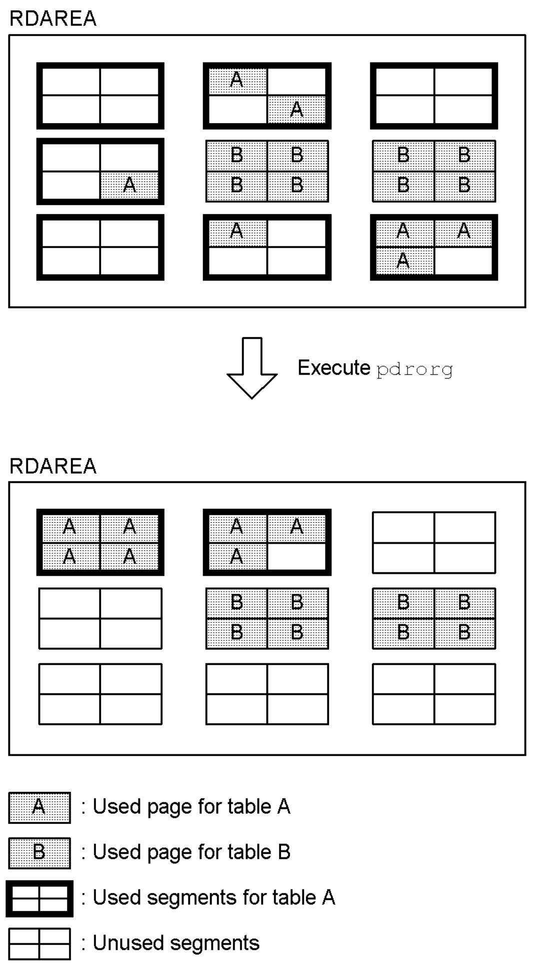
Before `pdreclaim` is executed, the deleted keys 2 and 5 are retained as remaining entries. Releasing these remaining entries and defragmenting their spaces into a contiguous space is called page compaction. Page compaction is also executed as an extension of SQL statements. However, when it is executed as an extension of SQL statements, online performance is adversely affected. You can avoid such adverse effects on online performance by executing page compaction in advance with `pdreclaim`.

If remaining entries are left as is, a lock-release wait or deadlock may occur. By executing page compaction with `pdreclaim`, you can also avoid lock-release waits and deadlock.

11.1.4 Difference in functions from pdrorg

The `pdrorg` utility should be used for reorganization when online operations can be suspended or when the inner replica facility can be used for reorganization in the online mode. This is because `pdrorg` provides more data rearrangement functions than `pdreclaim`. When `pdrorg` is used, data is rearranged as shown in Figure 11-6, thereby releasing more used free pages and used free segments.

Figure 11-6: When pdrorg is used to release used free pages and used free segments



Compared with `pdrorg`, `pdreclaim` provides the following advantages:

- You can reference and update a table and its indexes from a UAP while the utility is executing (except when the inner replica facility is used).
- There is no need for an unload data file or work files.

Therefore, if all of the following conditions are satisfied, we recommend that you use `pdreclaim`:

1. No variable-length character strings have been updated (no branch rows have

been created).

2. There have been no changes in the number of elements of repetition columns (no branch rows have been created).
3. No null values have changed to real data, or vice versa (no branch rows have been created).
4. No cluster key index is defined (there is no need to handle cluster keys).
5. There is no LOB column or column of an abstract data type with the LOB attribute.
6. During data deletions, a large amount of physically adjacent data was deleted at the same time (used free pages were created). Or, the table is a `SEGMENT REUSE` table.

A `SEGMENT REUSE` table with the `FIX` attribute for which no cluster key index is defined is an ideal candidate for `pdreclaim` processing. Even if these conditions are not all satisfied, `pdreclaim` can avoid the extreme performance degradation that can result from a free space search in used pages, because the utility releases used free pages. This could make it possible to change from weekly execution of `pdrorg` to monthly execution.

For indexes, use of `pdreclaim` is most suitable if deleted key values will never be re-registered. If key values are frequently updated and deleted, `pdreclaim` can reduce the frequency of space shortages in RDAREAs for indexes without having to reorganize the indexes with `pdrorg`.

To determine which utility should be used, check the results of executing the database condition analysis utility. If there are many pages whose usage is 0%, execute `pdreclaim`; if there are many pages whose usage is greatly different from the value specified for `PCTFREE` in the `CREATE TABLE` statement, execute `pdrorg`.

11.1.5 Execution environment

1. You can execute `pdreclaim` only while HiRDB is running.
2. Execute `pdreclaim` at the server machine that contains the single server or the system manager.
3. To execute `pdreclaim`, set the `LANG` environment variable. To use character codes that are not supported by the OS in an environment in which `pdreclaim` is executed, you must set the `PDLANG` environment variable. For details about `LANG` and `PDLANG`, see the *HiRDB Version 8 UAP Development Guide*.
4. You can execute `pdreclaim` on an RDAREA as long as it is in open status. If you specify the `-j` option, we recommend that you place the RDAREA in open and command shutdown status. For details about whether or not `pdreclaim` can be executed, see Appendix *C. RDAREA Status During Command Execution*.

5. You cannot execute multiple `pdreclaim` commands on the same table or indexes at the same one time (if executed, an error results). You can execute multiple `pdreclaim` commands concurrently on individual RDAREAs storing tables or indexes.

11.1.6 Executor

- When data dictionary tables are processed

  You need the `DBA` privilege.

- Other

  You need the `DBA` privilege (audit privilege if you are processing an audit trail table) or `INSERT` and `DELETE` privileges for the table to be processed.

When the Directory Server linkage facility is used, the indicated privilege must be assigned to the executor's specified authorization identifier or role.

11.2 Examples

This section presents four examples of `pdreclaim`.

(1) HiRDB/Single Server

Example 1:

This example releases used free pages in a table (`TBL1`). The example assumes that the table and indexes are defined as follows:

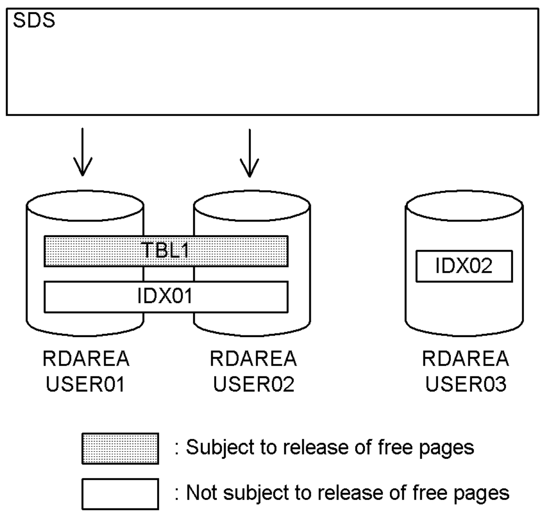
- Table definition

```
CREATE FIX TABLE TBL1(C1 INTEGER,C2 CHAR(10),C3 DEC(15))
    IN ((USER01) C1>0,(USER02))
```

- Index definitions

```
CREATE INDEX IDX01 ON TBL1(C1) IN ((USER01),(USER02))
CREATE INDEX IDX02 ON TBL1(C3) IN USER03
```

Overview


SDS
TBL1
IDX01
IDX02
RDAREA USER01
RDAREA USER02
RDAREA USER03
: Subject to release of free pages
: Not subject to release of free pages

- pdreclaim command

```
pdreclaim -k table -t TBL1
```

Explanation

`-k table`: Specifies that used free pages in a table are to be released.

`-t TBL1`: Specifies the table whose used free pages are to be released.

Example 2:

This example releases used free pages in all indexes defined for a table (`TBL1`). The example assumes that the table and indexes are defined as follows:

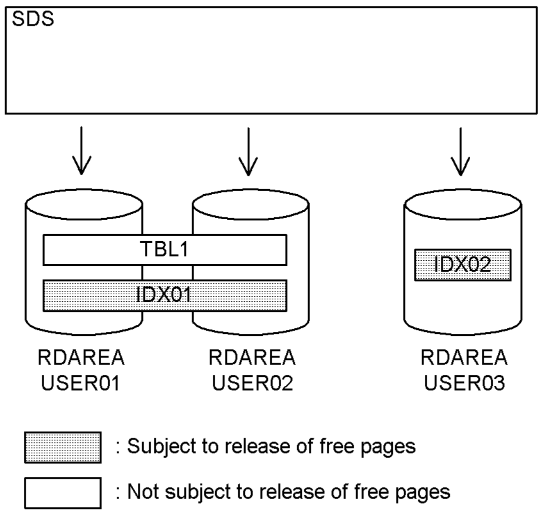
- Table definition

```
CREATE FIX TABLE TBL1(C1 INTEGER,C2 CHAR(10),C3 DEC(15))
    IN ((USER01) C1>0,(USER02))
```

- Index definitions

```
CREATE INDEX IDX01 ON TBL1(C1) IN ((USER01),(USER02))
CREATE INDEX IDX02 ON TBL1(C3) IN USER03
```

Overview


SDS
TBL1
IDX01
IDX02
RDAREA USER01
RDAREA USER02
RDAREA USER03
: Subject to release of free pages
: Not subject to release of free pages

- pdreclaim command

```
pdreclaim -k index -t TBL1
```

Explanation

`-k index`: Specifies that used free pages in indexes are to be released.

`-t TBL1`: Specifies the table whose used free pages are to be released.

(2) HiRDB/Parallel Server

Example 3:

This example releases used free pages in a table (`TBL1`). The example assumes that the table and indexes are defined as follows:

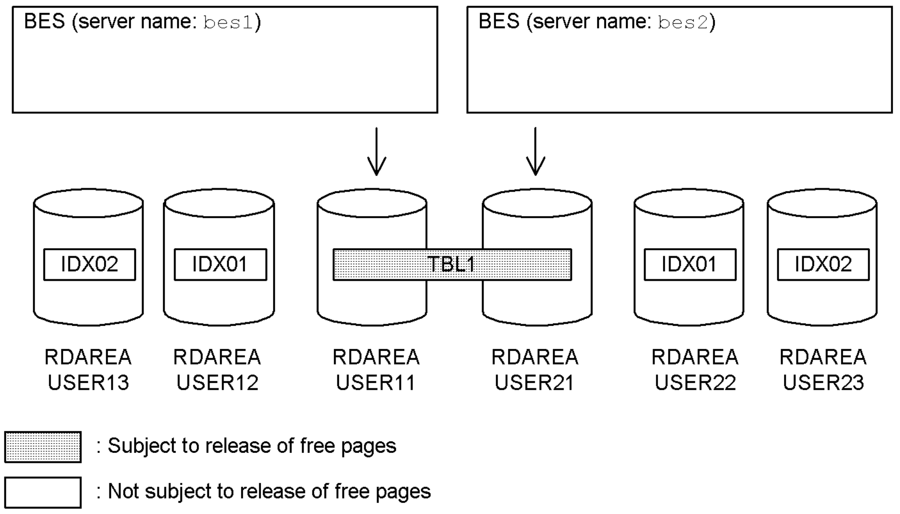
- Table definition

```
CREATE FIX TABLE TBL1(C1 INTEGER,C2 CHAR(10),C3 DEC(15))
    IN ((USER11) C1>0,(USER21))
```

- Index definitions

```
CREATE INDEX IDX01 ON TBL1(C1) IN ((USER12),(USER22))
CREATE INDEX IDX02 ON TBL1(C3) IN ((USER13),(USER23))
```

Overview



- pdreclaim command

```
pdreclaim -k table -t TBL1
```

Explanation

`-k table`: Specifies that used free pages in a table are to be released.

`-t TBL1`: Specifies the table whose used free pages are to be released.

Example 4:

This example releases used free pages in an index (IDX01) defined for a table (TBL1). The example assumes that the table and indexes are defined as follows:

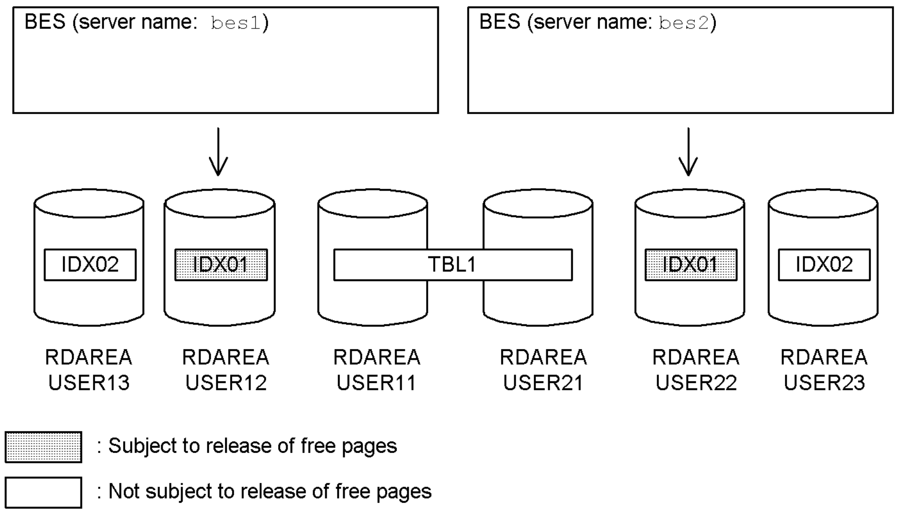
- Table definition

```
CREATE FIX TABLE TBL1(C1 INTEGER,C2 CHAR(10),C3 DEC(15))
    IN ((USER11) C1>0,(USER21))
```

- Index definitions

```
CREATE INDEX IDX01 ON TBL1(C1) IN ((USER12),(USER22))
CREATE INDEX IDX02 ON TBL1(C3) IN ((USER13),(USER23))
```

Overview



- pdreclaim command

```
pdreclaim -k index -t TBL1 control_file
```

Explanation

-k index: Specifies that used free pages in indexes are to be released.

-t TBL1: Specifies the table whose used free pages are to be released.

control_file: Control information file

- Contents of the control information file (control_file)

```
idxname name=IDX01    [1]
```

Explanation

1. Specifies the index whose used free pages are to be released.

Example 5:

This example releases used free pages from an index storage RDAREA (`USER12`) for an index (`IDX01`) defined for a table (`TBL1`). The example assumes that the table and indexes are defined as follows:

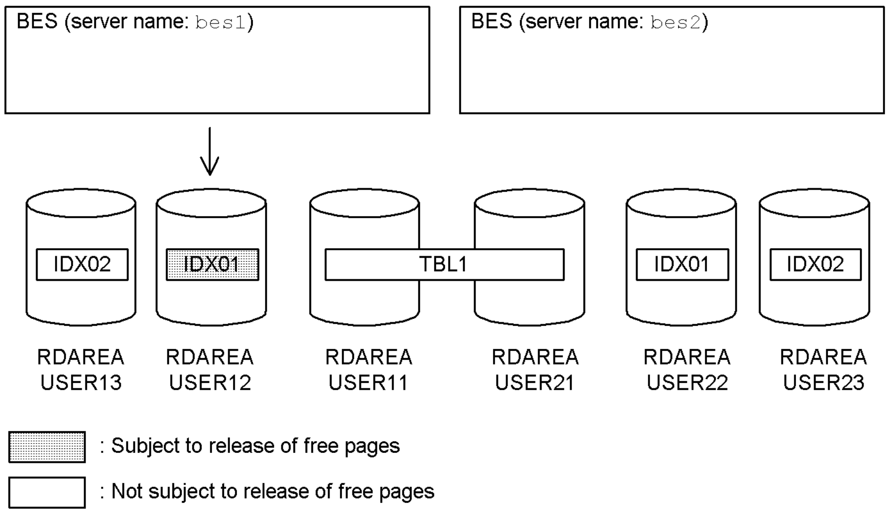
- Table definition

```
CREATE FIX TABLE TBL1(C1 INTEGER,C2 CHAR(10),C3 DEC(15))
    IN ((USER11) C1>0,(USER21))
```

- Index definitions

```
CREATE INDEX IDX01 ON TBL1(C1) IN ((USER12),(USER22))
CREATE INDEX IDX02 ON TBL1(C3) IN ((USER13),(USER23))
```

Overview



- pdreclaim command

```
pdreclaim -k index -t TBL1 control_file
```

Explanation

`-k index`: Specifies that used free pages in indexes are to be released.

`-t TBL1`: Specifies the table whose used free pages are to be released.

`control_file`: Control information file

- Contents of the control information file (`control_file`)

```
idxname name=IDX01 rdarea=USER12    [1]
```

Explanation

1. Specifies the index whose used free pages are to be released and its index storage RDAREA.

11.3 Command format

11.3.1 Format

This section describes the format of the `pdreclaim` command. In the following table, each number corresponds to the number assigned to each option.

No.	Format
1	`pdreclaim -k` *target-resource*
2	`-t {[`*authorization-identifier*`.]`*table-identifier*`|[`*authorization-identifier*`.]all}`
3	`[-r` *RDAREA-name*`]`
4	`[-c` *target*`]`
5	`[-m` *commit-interval*`]`
6	`[-u` *authorization-identifier*`]`
7	`[-w` *concurrently-executed-transaction-settlement-wait-time*`]`
8	`[-X` *response-monitoring-time-for-server-to-server-communication*`]`
9	`[-q` *generation-number*`]`
10	`[-j]`
11	`[-o]`
12	`(-x)`
13	`(-p)`
14	`[-n` *lock-retries-count*`]`
15	`[-s` *server-name*`]`
16	`[`*control-information-file-name*`]`

Note

If you specify *control-information-file-name*, be sure to specify it as the last option.

■ Relationships between options

Table 11-2 shows the relationships between options when `-c user` is specified, and Table 11-3 shows the relationships between options when `-c dic` is specified.

Table 11-2: Relationships between options when -c user is specified

Option	-k option		-t option	
	table	index	table-identifier	all
`-r` *RDAREA-name*	Y	N	Y	N
`-m` *commit-interval*	Y	Y	Y	Y
`-u` *authorization-identifier*	Y	Y	Y	Y
`-w` *concurrently-executed-transaction-settlement-wait-time*	Y	Y	Y	Y
`-X` *response-monitoring-time-for-server-to-server-communication*	Y	Y	Y	Y
`-q` *generation-number*	Y	Y	Y	Y
`-j`	Y	Y	Y	Y
`-o`	Y	N	Y	Y
`-x`	N	Y	Y	Y
`-p`	Y	Y	Y	Y
`-n` *lock-retries-count*	Y	Y	Y	Y
`-s` *server-name*	Y	Y	Y	Y
control-information-file-name	Y	Y	Y	Y

Legend:

Y: Can be specified

N: Cannot be specified

Table 11-3: Relationships between options when -c dic is specified

Option	-k option	
	table	index
`-t {[`*authorization-identifier*`.]`*table-identifier*`|[`*authorization-identifier*`.]all}`	N	N
`-r` *RDAREA-name*	N	N
`-m` *commit-interval*	Y	Y
`-u` *authorization-identifier*	Y	Y
`-w` *concurrently-executed-transaction-settlement-wait-time*	Y	Y

Option	-k option	
	table	index
`-X` *response-monitoring-time-for-server-to-server-communication*	Y	Y
`-q` *generation-number*	N	N
`-j`	Y	Y
`-o`	Y	N
`-x`	N	Y
`-p`	Y	Y
`-n` *lock-retries-count*	Y	Y
`-s` *server-name*	N	N
control-information-file-name	Y	Y

Legend:

Y: Can be specified

N: Cannot be specified

11.3.2 Options

(1) -k target-resource

Specifies the type of resource (table or index) whose used free pages are to be released.

`table`:

Specifies that a table is to be processed.

`index`:

Specifies that indexes are to be processed.

(2) -t {[authorization-identifier.]table-identifier|[authorization-identifier.]all}

~ <identifier>

Specifies the name of the table whose used free pages are to be released.

If you omit the authorization identifier, the utility assumes the user name used to connect to HiRDB.

If you specify `all`, the utility processes all tables or indexes that belong to the schema whose name is the specified authorization identifier. In this case, you cannot specify the `idxname` statement.

(3) -r RDAREA-name

~ <identifier>

Specifies the name of the RDAREA that is to be processed (among the RDAREAs storing the table specified with the `-t` option).

You can specify this option only when `-k table` is specified. If you specify `-k index`, specify the `idxname` statement.

You can specify only a user RDAREA. Batch specification of RDAREA names is not permitted.

(4) -c target

~ <<user>>

Specifies the type of RDAREA that is to be the target of this processing (user RDAREA or data dictionary RDAREA).

`user`:

Specifies that a user RDAREA is to be processed.

`dic`:

Specifies that a data dictionary RDAREA is to be processed.

(5) -m commit-interval

~ <unsigned integer> ((0-100000)) <<1000>>

Specifies the transaction settlement interval for the used free page release processing, expressed as the number of pages released. When the number of released pages per RDAREA for the table or index reaches the specified value, the transaction is settled. The utility executes only page compaction on the released pages, including pages that could not become used free pages.

If you specify `0`, the transaction settles when all used free pages in the corresponding RDAREA have been released.

By specifying a large value, you can reduce the amount of transaction log information. However, `pdreclaim` becomes a long-running transaction, losing the timing of synchronization point validation.

(6) -u authorization-identifier

Specifies the authorization identifier of the user who executes `pdreclaim`.

For details about the authorization identifier, see *8.9.2(10) -u authorization-identifier*.

(7) -w concurrently-executed-transaction-settlement-wait-time

~ <unsigned integer> ((0-3600)) <<0>>

Specifies a wait time for `pdreclaim` (in seconds). When a UAP or another utility accesses a table or index while its free pages or free segments are being released, `pdreclaim` is placed in lock-release wait status or in transaction settlement wait status. This option specifies that wait time.

`pdreclaim` is placed in this wait status in the following situations:

- When `pdreclaim` is placed in lock-release wait status

  `pdreclaim` is placed in lock-release wait status when either of the following occurs:

  - A UAP or another utility is accessing the table or index whose free segments are to be released
  - When free pages are being released, transaction processing that places the RDAREA in the EX lock mode is executing: for instance, data loading by RDAREA is underway or a UAP using the local buffer is executing.

- When `pdreclaim` is placed in transaction settlement wait status

  If a UAP is updating the index page that is to be released, `pdreclaim` is placed in UAP transaction settlement wait status.

Table 11-4 shows the relationship between the `-w` option specification and `pdreclaim`'s wait time.

Table 11-4: Relationship between -w option specification and pdreclaim's wait time

Processing	Specification of -w option	Transaction settlement wait time	Lock-release wait time per resource
Releasing free pages in a table	Yes	—	`-w` option value
	No	—	Waits indefinitely until the lock status is released
Releasing free pages in an index	Yes	`-w` option value	`-w` option value
	No	Waits indefinitely until the other transaction is settled	Waits indefinitely until the lock status is released
Releasing free segments in a table	Yes	—	`-w` option value
	No	—	Waits indefinitely until the lock status is released
Releasing free segments in an index	Yes	—	`-w` option value
	No	—	Waits indefinitely until the lock status is released

Legend:

— : Not applicable because it has nothing to do with the transaction settlement wait status

Rules

1. If the UAP's transaction is not settled within the specified amount of time after `pdreclaim` is placed in the wait status, `pdreclaim` cancels processing with return code `4`. If the lock-release wait status is not released, `pdreclaim` cancels processing with return code `8`.
2. If the `-w` option is omitted or `0` is specified in the `-w` option, `pdreclaim` waits without monitoring until the UAP's transaction is settled or the lock is released.

(8) -X response-monitoring-time-for-server-to-server-communication

~ <unsigned integer> ((1-65535)) <<300>>

If an error, such as a communication error, occurs at the server where the command was executed, the command may stop responding and the application may stop. To detect errors, `pdreclaim` enables you to monitor a response time during communication for dictionary manipulation that is performed by the command.

You set in the `-X` option the response monitoring time during dictionary manipulation (in seconds). If the execution time during dictionary manipulation exceeds the value set in the `-X` option, `pdreclaim` treats it as a dictionary access error and cancels processing with return code 8.

Criteria

- If you want to detect an error in less time than 300 seconds in the event of a no-response from the server due to a communication error or unit down, specify a smaller value than 300 in the `-X` option.
- If the system switchover facility is used, the command may keep waiting for a response even though system switchover has been completed. In such a case, you can terminate the command immediately by reducing the monitoring time.
- The specified monitoring time may result in a timeout if a response from the dictionary is delayed and if the utility's preprocessing is not completed within 300 seconds, which is the default value of the `-X` option. This can happen when many applications and utilities are executing concurrently. In such an environment, specify a value greater than 300 in the `-X` option.

(9) -q generation-number

~ <unsigned integer> ((0-10))

When the inner replica facility is used, this option specifies the generation number of the target RDAREA.

Specify the generation number as follows:

`0`: Processes the original RDAREA.

`1` to `10`: Processes the replica RDAREA with the specified generation number.

Criteria

Specify this option to process an RDAREA other than the current RDAREA using the inner replica facility.

Rules

1. When the inner replica facility is not used, this option is disabled.
2. When this option is omitted, the current RDAREA is assumed.
3. If a replica RDAREA is to be processed, specify the name of the original RDAREA in the `-r` option and the target generation number in the `-q` option.
4. `pdreclaim` checks the target RDAREA to determine whether its generation number matches the specification. Table 11-5 shows the RDAREAs whose generations are checked by `pdreclaim`.

Table 11-5: RDAREAs subject to generation checking by pdreclaim

Type of processing		**Target RDAREA**		**Specification of option or in control information file**
		For storing a table	**For storing an index**	
Releasing used free pages in table	In units of tables	Y	—	`-k table`
	In units of RDAREAs	Y	—	`-k table -r` *RDAREA-name*
Releasing used free pages in index	In units of indexes	Y	Y	`-k index`
	In units of servers	Y	Y	`-k index` `idxname` statement (with `server` specified)
	In units of RDAREAs	Y	Y	`-k index` `idxname` statement (with `rdarea` specified)

Legend:

Y: Checked

— : Not checked

(10) -j

Specifies that if there is a used free segment and all its pages become unused, the used free segment is to be released (to create an unused segment).

Criteria

When you specify the `-j` option, the table or index being processed becomes inaccessible. Therefore, use this option as explained below:

1. During daytime online operation, execute `pdreclaim` without specifying `-j`.
2. During nighttime batch operation, execute `pdreclaim` with `-j` specified.

Rules

When you specify the `-j` option, the released unused segments become available to any table or index, whether or not it has been using the segments. However, if you specify `-k table` or `-k index`, the table storage RDAREA or index storage RDAREA is placed in the EX lock mode, in which case `pdreclaim` can no longer be concurrently executable with a UAP.

(11) -o

Specifies that free pages are to be released in the table, but that page compaction is not to be executed.

When page compaction is executed to release free space in a page that resulted from updating and deleting rows, database update log information (FJ) is output and its volume is proportional to the amount of free space in the page that is released. You specify the `-o` option in order to reduce the volume of this log output. You specify this option even when the purpose is to release free pages whose usage rate is 0%. This specification can reduce `pdreclaim`'s execution time.

(12) -x

Specifies that page compaction is to be executed in order to release remaining entries in a unique index when free index pages are released. By executing page compaction, you can maintain online performance and avoid lock-release wait and deadlock.

When page compaction is executed to release remaining entries, database update log information (FJ) is output and its volume is proportional to the amount of key space that is released.

Note that page compaction is executed only when the index to be released is a unique index. For any other type of index, the `-x` option is ignored.

Criteria

Specifying the `-x` option results in overhead for page compaction. If you want to reduce the volume of the database update log information (FJ) or if performance is more important, do not specify the `-x` option.

(13) -p

Specifies that the update buffer is to be applied to the database at the point of commit. If you specify the `-p` option when you execute the utility at the same time as you execute a UAP, you can reduce the workload of synchronization point processing, thereby reducing the effects on the performance of other transactions. Note that the performance of `pdreclaim` is lower than when the `-p` option is omitted.

Specify the `-m` option to set the commit interval in such a manner that a commit is acquired before a synchronization point dump is acquired.

There is a relationship between the `pdreclaim` processing and the `pd_max_commit_write_reclaim_no` operand value in the system common definition. Table 11-6 describes the `pdreclaim` processing when the `pd_max_commit_write_reclaim_no` operand and the `-p` option are specified.

Table 11-6: pdreclaim processing when the pd_max_commit_write_reclaim_no operand and the -p option are specified

pd_max_commit_write_reclaim_no operand specification		pdreclaim processing when the -p option is specified
Specified	0	`pdreclaim` terminates with an error.
	Other than 0	There can be as many concurrent executions per server of the `pdreclaim` utility with the `-p` option specified as the value specified in the `pd_max_commit_write_reclaim_no` operand. Any excess `pdreclaim` utility executions terminate with an error.
Omitted	`v7compatible` or `v6compatible` is specified in the `pd_sysdef_default_option` operand	`pdreclaim` terminates with an error.
	Other	There can be up to 10 concurrent executions per server of the `pdreclaim` utility with the `-p` option specified. Any excess `pdreclaim` utility executions terminate with an error.

(14) -n lock-retries-count

~ <unsigned integer> ((0-3600)) <<0>>

Specifies the maximum number of lock acquisition requests to be released in the case of releasing free segments. Because the `-n` option is applicable only when free segments are released, you must also specify the `-j` option when you specify the `-n` option.

The following describes release of lock acquisition requests in order to release free segments.

If you specify the `-j` option to release free segments, the target RDAREA is placed in the EX lock mode. If an attempt is made to release free segments in an RDAREA that is being processed by another transaction, the processing may be placed in lock-release wait status.

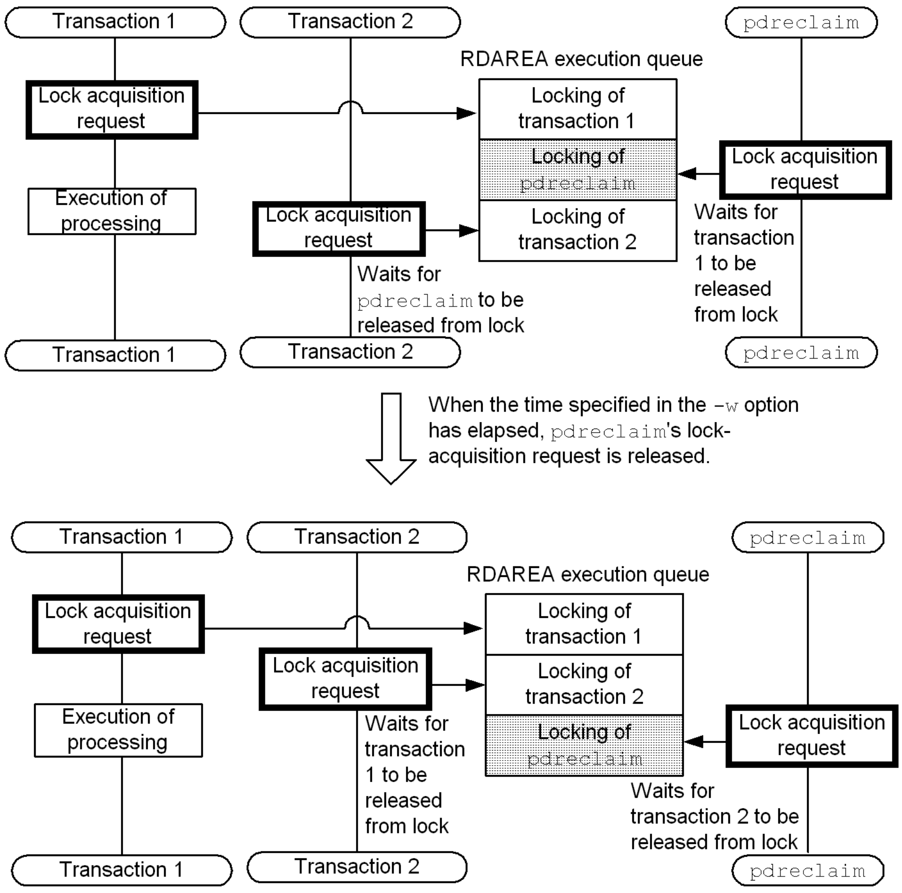
If another transaction acquires a lock on the RDAREA while the free segment release processing is in lock-release wait status, the first transaction that acquired a lock on the RDAREA and the free segment release processing are both placed in wait status until the lock is released.

When the `-n` option is specified, the lock acquisition request for the free segment release processing is released when the time specified in the `-w` option has elapsed, thereby giving preferences to the other transactions' lock-release wait status. The free segment release processing waits for lock release for the amount of time specified in the `-w` option. Therefore, if you specify the `-n` option, you must also specify the `-w` option.

If free segments are not released, the lock acquisition request for free segment release processing is cancelled and then issued again.

Figure 11-7 shows the relationship between the lock acquisition request and the lock-release wait queue.

Figure 11-7: Relationship between lock acquisition request and lock-release wait queue



Note: You specify in the `-n` option the maximum retries count during the flow of a lock acquisition request for free segment release processing, lock-release wait, and release of the lock acquisition request for free segment release processing.

When the `-n` option is specified, the utility waits for lock release for as long as the amount of time obtained from *-w option value* x *-n option value*. If this value exceeds the `pdreclaim` execution monitoring time specified in the `exectime` operand in the `option` statement, `pdreclaim` displays the `KFPL11111-E` message while waiting for lock release and terminates itself forcibly. It is important that you specify an appropriate value in the `-n` option, taking into account the following:

Relationship between the `-n` option and the `pdreclaim` execution monitoring time

$A + B + C < D \times 60$

Legend:

A: Transaction settlement wait time (`-w` option value)

B: Lock-release wait time for the free segment release processing (`-w` option value x `-n` option value)

C: Execution time for free segment release processing

D: `pdreclaim` execution monitoring time specified in the `exectime` operand of the `option` statement

(15) -s server-name

~ <identifier>

This option is applicable to a HiRDB/Parallel Server only (it is ignored if specified for a HiRDB/Single Server).

Specifies the name of the back-end server that is to manage execution of `pdreclaim`.

Criteria

When this option is omitted, one of the back-end servers is selected automatically; therefore, normally there is no need to specify this option.

Specify this option if you execute many `pdreclaim` commands concurrently and the number of utility processes per server (`0mrorg`) is 32 or greater. You can determine the value of `0mrorg` by checking the result of the `pdls -d prc` command for the number of process IDs (`0mrorg`).

(16) control-information-file-name

~ <path name>

Specifies the name of the control information file that contains the control statements for `pdreclaim`.

You can specify the following control statements in the control information file; for details about each control statement, see Sections *11.3.3* and *11.3.4*:

- `idxname` statement
- `option` statement

11.3.3 idxname statement (specification of index information)

The `idxname` statement specifies information about an index from which used free pages are to be released (`-k index`).

Criteria

When `-k index` is specified, the utility processes all indexes defined for the table that is specified with the `-t` option. You specify the `idxname` statement to process only a particular index, server, or RDAREA.

Rules

1. Specify the `name` operand before the `server` and `rdarea` operands.
2. Enclose an identifier in double quotation marks (`"`) if it contains any spaces or lowercase letters. Lowercase letters are treated as uppercase letters, if not enclosed in double quotation marks.
3. Once you specify an `idxname` statement with the `rdarea` operand specified, you cannot specify another `idxname` statement specifying only the `name` operand or an `idxname` statement specifying the `server` operand.
4. You cannot specify more than one `idxname` statement specifying the same index identifier, except when the `rdarea` operand is specified.

(1) Format

```
idxname name={index-identifier|*}
         [{server=server-name[,server-name]...
             |rdarea=RDAREA-name[,RDAREA-name]...}]
```

(2) Explanation

(a) name={index-identifier|*}

index-identifier ~ <identifier>

Specifies the identifier of the index whose used free pages are to be released.

There is no need to specify the authorization identifier, because the utility assumes the authorization identifier of the table specified with the `-t` option.

`*` Specifies that used free pages are to be released from all indexes defined for the table.

The system processes all indexes for the table both when the `idxname` statement specifying `*` (without the `server` operand) is specified and when the `idxname` statement is omitted. If you specify the `idxname` statement specifying both `*` and the `server` operand, the utility processes only those indexes located at the back-end server specified in the `server` operand.

When `*` is specified, there can be only one `idxname` statement.

(b) server=server-name[,server-name]...

~ <identifier>

This operand is applicable to a HiRDB/Parallel Server only. Specifies the name of the server (back-end server name) when only the part of the index specified in the `name` operand that is located at the specified server is to be processed. When this operand is omitted, the utility assumes all servers that contain the index.

(c) rdarea=RDAREA-name[,RDAREA-name]...

~ <identifier>

Specifies the names of RDAREAs for the index that is specified in the `name` operand. When this option is omitted, the utility assumes all RDAREAs that store the index. This operand cannot be specified when an asterisk (`*`) is specified in the `name` operand.

11.3.4 option statement (specification of optional information)

The `option` statement specifies `pdreclaim`'s execution monitoring time.

Criteria

Specify the `option` statement to monitor the execution time of `pdreclaim`.

Rules

The utility ignores any operand other than `exectime` (such as those permitted in the `option` statement of `pdrorg`), if specified. Specifying other operands will not result in a control statement error.

(1) Format

```
option exectime=pdreclaim-monitoring-time
```

(2) Explanation

(a) exectime=pdreclaim-monitoring-time

~ <unsigned integer> ((0-35791394))

Specifies the `pdreclaim` execution monitoring time in minutes.

Guideline for the specification value

The purpose of this operand is to detect a no-response error, not to monitor the execution time of a long transaction. Therefore, you should specify a sufficient time for processing the applicable table. For example, to monitor the execution time of a `pdreclaim` that should terminate in 7-8 minutes, specify `exectime=20`, not `exectime=10`.

Rules

1. If you omit this operand or specify `0`, the system will not monitor the execution time.
2. This operand takes the precedence over the `pd_utl_exec_time` operand in the system definition. Table 11-7 describes the relationship between the `exectime` operand and the `pd_utl_exec_time` operand in the system definition.

Table 11-7: Relationship between the exectime operand and the pd_utl_exec_time operand in the system definition (pdreclaim)

exectime operand value	pd_utl_exec_time operand value in the system definition	
	Omitted or 0	A
Omitted	—	Monitoring with value A
0	—	—
B	Monitoring with value B	Monitoring with value B

Legend:

—: Execution time is not monitored.

3. If `pdreclaim` processing does not terminate within the specified time, terminate the utility process forcibly and collect troubleshooting information to determine the cause of the no-response error. Note that this statement monitors the execution time of the utility process that releases used free pages (`0mrorg`) and the single server (or back-end server). If an error occurs in the command process (`pdreclaim` or `pdrorg`), the process cannot be terminated forcibly. For details about the error information to be collected, see the `pd_utl_exec_time` operand in the manual *HiRDB Version 8 System Definition*.

11.4 pdreclaim processing results

If the return code from execution of `pdreclaim` is `0` or `4`, the `pdreclaim` processing results are output to the standard output.

(1) Processing results of pdreclaim (with -j not specified) (releasing used free pages in a table)

```
pdreclaim VV-RR        *** DB RECLAIM ***       YYYY-MM-DD
hh:mm:ss[1]

TABLE NAME : USERA.TBL01 [2]
[3]     [4]                                  [5]          [6]
No.  RDAREA NAME                        PAGE COUNT  SEGMENT COUNT
  1  USER01                                 11,395            244
  2  USER02                                 11,403            252

14284 16:22:54 un01    KFPL00739-I Pdreclaim terminated, return
code=0
```

Explanation:

1. Header for the processing results

   VV-RR: Version, revision number

   YYYY-MM-DD hh:mm:ss: `pdreclaim` start date and time
2. Name of the table that was processed

   If the processing was in units of schemas, the information beginning with this item is repeated.
3. Serial numbers
4. Names of the table storage RDAREAs
5. Numbers of pages released
6. Numbers of segments that can be released

   Each RDAREA's number of segments that can be released includes not only segments that can be changed to unused segments by releasing used free pages by `pdreclaim`, but also segments that can be changed to unused segments by the `DELETE` statement for a table on which the `LOCK` statement was executed (executing the `DELETE` statement in the no-log mode automatically places the segments in `LOCK`-executed status).

(2) Processing results of pdreclaim (with -j specified) (releasing used free segments in a table)

```
pdreclaim VV-RR        *** DB RECLAIM ***        YYYY-MM-DD 
hh:mm:ss[1]

TABLE NAME : USERA.TBL01 [2]
[3]      [4]                              [5]           [6]
No.  RDAREA NAME                    PAGE COUNT  SEGMENT COUNT
  1  USER01                                  -            244
  2  USER02                                  -            252

16066 16:23:57 un01   KFPL00739-I Pdreclaim terminated, return 
code=0
```

Explanation:

1. Header for the processing results

   VV-RR: Version, revision number

   YYYY-MM-DD hh:mm:ss: `pdreclaim` start date and time
2. Name of the table that was processed

   If the processing was in units of schemas, the information beginning with this item is repeated.
3. Serial numbers
4. Names of the table storage RDAREAs
5. Always - when the `-j` option was specified
6. Numbers of segments released

(3) Processing results of pdreclaim (with -j not specified) (releasing used free pages in indexes)

```
pdreclaim VV-RR        *** DB RECLAIM ***        YYYY-MM-DD 
hh:mm:ss[1]

INDEX NAME : USERA.(PRIMARY0000131193) [2]
[3]      [4]                              [5]           [6]
No.  RDAREA NAME                    PAGE COUNT  SEGMENT COUNT
  1  USER01                                 32              0
  2  USER02                                 32              0

INDEX NAME : USERA.IDX01
No.  RDAREA NAME                    PAGE COUNT  SEGMENT COUNT
```

```
  1  USER01                                     815            15
  2  USER02                                     815            15

INDEX NAME : USERA.IDX02
No.  RDAREA NAME                          PAGE COUNT  SEGMENT COUNT
  1  USER01                                      93             1

 1273 16:58:26 un01    KFPL00739-I Pdreclaim terminated, return
code=0
```

Explanation:

1. Header for the processing result

   VV-RR: Version, revision number

   YYYY-MM-DD hh:mm:ss: `pdreclaim` start date and time
2. Name of the index that was processed

   If multiple indexes were processed, the information beginning with this item is repeated.
3. Serial numbers
4. Names of the index storage RDAREAs
5. Numbers of pages released
6. Numbers of segments that can be released

   Each RDAREA's number of segments that can be released includes not only segments that can be changed to unused segments by releasing used free pages by `pdreclaim`, but also segments that can be changed to unused segments by the `DELETE` statement for a table on which the `LOCK` statement was executed (executing the `DELETE` statement in the no-log mode automatically places the segments in `LOCK`-executed status).

(4) Processing results of pdreclaim (with -j specified) (releasing used free segments in indexes)

```
pdreclaim VV-RR        *** DB RECLAIM ***        YYYY-MM-DD
hh:mm:ss[1]

INDEX NAME : USERA.(PRIMARY0000131193) [2]
[3]      [4]                                [5]           [6]
No.  RDAREA NAME                          PAGE COUNT  SEGMENT COUNT
  1  USER01                                       -             0
  2  USER02                                       -             0

INDEX NAME : USERA.IDX01
```

```
No.  RDAREA NAME                        PAGE COUNT  SEGMENT COUNT
  1  USER01                                      -             15
  2  USER02                                      -             15

INDEX NAME : USERA.IDX02
No.  RDAREA NAME                        PAGE COUNT  SEGMENT COUNT
  1  USER01                                      -              1

 1381 16:59:34 un01    KFPL00739-I Pdreclaim terminated, return
code=0
```

Explanation:

1. Header for the processing results

   VV-RR: Version, revision number

   YYYY-MM-DD hh:mm:ss: `pdreclaim` start date and time

2. Name of the index that was processed

   If multiple indexes were processed, the information beginning with this item is repeated.

3. Serial numbers

4. Names of the index storage RDAREAs

5. Always - when the `-j` option was specified

6. Numbers of segments released

(5) Notes on the processing results

The processing results are output in the order that processing of the resources was completed.

When the `-j` option is specified, the numbers of pages released as displayed by `pdreclaim` may not match the numbers of pages that can be released (`Collect Prearranged Page`) that is displayed as the analysis result of `pddbst`. Also, the numbers of segments that can be released (processing results of `pdreclaim` without the `-j` option specified) may not match the numbers of segments released (processing results of `pdreclaim` with the `-j` option specified). The reasons are explained below.

(a) When a page is expected to be releasable, but pdreclaim is unable to release it

- To allow concurrent execution of a UAP and utility update processing, `pdreclaim` does not release any used free page that may result in concurrent updating. Also, the utility releases used free pages in such a manner that the number of index levels is not reduced.
- If `pdrorg` or `pdrorg` with the synchronization point specification terminates

abnormally, `pdreclaim` does not release the first segment allocated to the corresponding table even if that segment contains no data.

- If `pdreclaim` is executed concurrently with update processing, a used free page or segment that was determined to be releasable may be used for update processing.

(b) When a page is expected to be nonreleasable, but pdreclaim can release it

- An intermediate page of an index is released when all of its lower leaf pages are released by `pdreclaim`. However, if `pddbst` is executing, the usage of the intermediate page cannot be treated as being 0% because there are leaf pages.

11.5 Notes

1. The maximum number of `pdreclaim` commands that can be executed concurrently is the same as for the `pdrorg` command because `pdreclaim` executes `pdrorg` internally. Specifically, the maximum number of `pdreclaim` commands that can be executed concurrently equals the maximum number of `pdrorg` commands that can be executed concurrently minus the number of `pdrorg` commands that are currently executing.
2. `pdreclaim` cannot process view tables or external tables.
3. `pdreclaim` cannot process LOB RDAREAs. This means that `pdreclaim` cannot process any LOB RDAREA containing an abstract data type or plug-in index.
4. If a UAP accesses a page in an index that is being released, `pdreclaim` is placed in wait status until the corresponding process transaction is settled. You can specify a wait time for `pdreclaim` in the `-w` option.
5. To terminate `pdreclaim` forcibly, use the `pdcancel` command. Other commands such as the OS's `kill` command cannot terminate `pdreclaim`.
6. When executing `pdreclaim` with the `-j` option specified, we recommend that you shut down the applicable RDAREAs with the `pdhold` command to protect UAPs from being placed in lock-release wait status unnecessarily.

   When the `-j` option is specified in `pdreclaim`, the RDAREA storing the specified table and index is placed in the EX lock mode, making the entire RDAREA inaccessible. This lock control functions based on the specification of the `-w` and `-n` options, as shown in Table 11-8.

Table 11-8: Lock control during free segment release processing

Option value		**Lock-release wait time per processing**	**In the event of a lock error**			
-w	**-n**		**Lock-release wait timeout error**			**Other error**
			Retry locking	**Retries count**	**Processing**	**Processing**
Not specified	—	Indefinite	—	—	—	—

Option value		Lock-release wait time per processing	In the event of a lock error			
			Lock-release wait timeout error			Other error
-w	-n		Retry locking	Retries count	Processing	Processing
Specified	Not specifi ed	Time specified in the `-w` option	No	0	Displays an error message and then cancels processing.	Processing
	Specifi ed		Yes	Count specified in the `-n` option	Retries locking as many times as the value specified in the `-n` option. If all retries result in a timeout, the system displays an error message and then cancels processing.	

Legend:

—: Not applicable

Table 11-9 shows the resources that are locked during free segment release processing.

Table 11-9: Resources that are locked during free segment release processing

Resource name	Resource number	Node	Purpose	Whether or not locking is retried
`pdreclaim`	`5006`	`EX`	Suppression of multiple concurrent executions of `pdreclaim` on the same resource	There are no retries because processing is cancelled if `pdreclaim` is executing.

Resource name	Resource number	Node	Purpose	Whether or not locking is retried
RDAREA ID	`0001`	`EX`	Suppression of access to an RDAREA that is the subject of free segment release processing	Locking is retried when a lock-release wait timeout error occurs and the `-n` option is specified.
RRAMB	`0102`	`SR`		No timeout error occurs because this resource is acquired only after the RDAREA has been placed in the EX lock mode. Therefore, there are no retries.

7. Table 11-10 shows the return codes of `pdreclaim`.

Table 11-10: Return codes of pdreclaim

Return code	Description	Action
`0`	Release of used free pages or segments has been completed.	None
`4`	Processing was cancelled because a timeout occurred while waiting for settlement of a UAP transaction.	The table and index status is guaranteed because release of used free pages was cancelled due to a timeout. The table and indexes can be accessed from UAPs as is. To release the remaining used free pages, re-execute `pdreclaim`.
`8`	Processing terminated abnormally.	The table and index status is guaranteed because release of used free pages was cancelled due to an error. Even after abnormal termination, the table and indexes can be accessed from UAPs. To re-execute `pdreclaim`, first check the displayed error messages and eliminate the cause of the error.

8. If the target table contains a user-defined column of an abstract data type, `pdreclaim` cannot be executed.
9. To execute `pdreclaim` on a shared table or shared index, the system places the RDAREAs containing the shared table or shared index in the EX lock mode. If the corresponding RDAREAs contain other tables or indexes, these tables and indexes cannot be referenced or updated. For details about the lock mode used for executing `pdreclaim`, see Appendix *B.2 Lock mode for utilities*.
10. For a HiRDB/Parallel Server, if a transmission message from a back-end server to `pdreclaim` is delayed for 1 second or more due to traffic on the communication line, the corresponding back-end server's processing result may not be displayed at the standard output. In this case, check the `KFPL00714-I` message that is output to the message log file or `syslogfile` to determine the

processing result.

11. Do not execute a definition SQL statement on a table or index that is being processed by `pdreclaim`. If a definition SQL statement is so executed, `pdreclaim` terminates abnormally. If `pdreclaim` is executing in units of schemas, it processes all tables and indexes that belong to the corresponding schema.
12. If you selected `utf-8` as the character encoding in the `pdsetup` command, you can use a file with a BOM as the control statements file for `pdreclaim`; however, the BOM is skipped.

Chapter

12. Global Buffer Residence Utility (pdpgbfon)

This chapter explains the global buffer residence utility (`pdpgbfon`) that can be used to read data pages, such as table data pages and index pages, into a global buffer immediately after HiRDB starts (before starting online jobs, for example).

12.1 Overview

12.1.1 Overview of pdpgbfon

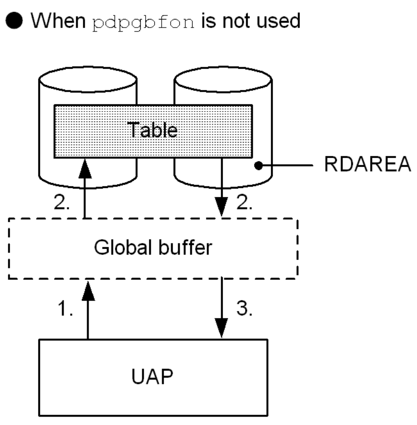
When a database is accessed, finding the desired target data in a global buffer without having to perform any physical input/output operation is called a *buffer hit*. When the probability of a buffer hit (buffer hit rate) is high, database access performance is stable. When HiRDB has just started, there is no information about tables and indexes in global buffers, so the buffer hit rate is low, resulting in unstable database access performance.

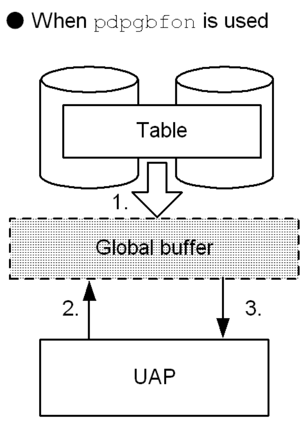
The global buffer residence utility (`pdpgbfon`) reads information about tables and indexes into global buffers in advance. If you execute `pdpgbfon` as soon as HiRDB has started, such as before startup of online jobs, you can expect stable database access performance.

For example, in an environment where all tables and indexes can be read from the database to a global buffer, high-speed database operations can be achieved without any physical input/output operations.

Figure 12-1 shows the effects of `pdpgbfon`.

Figure 12-1: Effects of pdpgbfon





Explanation

When pdpgbfon is not used:

1. UAP accesses the table.
2. Page information is read from the table because the global buffer does not contain the page information (physical input/output operation occurs).

3. The result is returned. Whenever this page information is accessed again subsequently, a table read operation will not be necessary because the page has been read into the global buffer. When page information from another page is accessed, the read operation of step 2 will be required.

When pdpgbfon is used:

1. `pdpgbfon` is executed, reading table page information into the global buffer.
2. UAP accesses the table.
3. The result is returned without reading the page information from the table because the page information is already in the global buffer. Whenever this table is accessed again subsequently, a read operation on the table will not be necessary (no physical input/output operation will occur).

12.1.2 Functions of pdpgbfon

(1) Overview of functions

`pdpgbfon` is used to improve the buffer hit rate, even immediately after startup of online jobs. It does this by reading the applicable pages of resources (tables or indexes) into global buffers. Once an applicable page has been placed in a global buffer, page read operations are not needed.

To execute `pdpgbfon`, there must be more global buffer sectors than the number of pages used to store the target resource (before execution of `pdpgbfon`, enough global buffer space to store all the data for the target resource must have been allocated).

Because `pdpgbfon` pre-reads data in the order that the pages are stored, the prefetch facility is enabled. When you define the global buffer, you can reduce execution time by specifying the prefetch count.

If there are not enough global buffer sectors, the LRU management method removes the oldest page information from the global buffer (the page in the global buffer that was accessed least recently according to the `pd_dbbuff_lru_option` operand value in the system definition). Therefore, executing `pdpgbfon` serves no purposes when there are not enough global buffer sectors. You can use the `pdbufls` command to check the number of global buffer sectors and the database condition analysis utility (`pddbst`) to check the number of pages being used to store the target resource.

To determine whether or not `pdpgbfon` read all pages of the target resource into the global buffer, you check the `pdpgbfon` return code or a processing result output to the standard output (reference buffer flush count). Note that the `pdpgbfon` processing result contains only information about the resource processed by `pdpgbfon`. To obtain all the global buffer processing results and global buffer status details, use the `pdbufls` command.

(2) Operating method

Figure 12-2 shows an example of the `pdpgbfon` operating method.

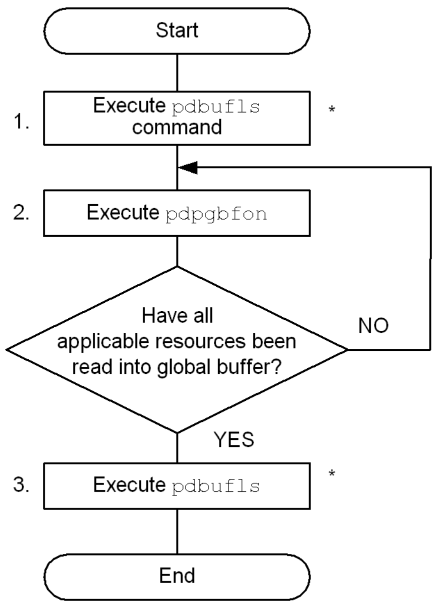


Figure 12-2: Example of pdpgbfon operating method

* This step need not be executed when the number global buffer sectors is greater than the number of storage pages.

Explanation

1. `pdbufls` command is executed to display global buffer statistical information. The purpose is to initialize the statistical information (the displayed information is not used).

2. `pdpgbfon` is executed as many times as there are resource elements to be read into the global buffer (table data pages or index pages). If the `pdpgbfon` return code is `0` every time is executes, there is no need to execute step 3.

3. `pdbufls` command is executed again to display global buffer statistical information. If the displayed reference buffer flush count (`RFFLS`) or update buffer flush count[1] (`UPFLS`) is not `0`, some pages have been swept out of the global buffer. In this case, either increase the size of the global buffer or consider reorganization of the table.[2] Note that the reference GET count (`REFGET`) is greater than the actual number of data storage pages because it includes the management page re-read count.

[1] `pdpgbfon` does not update data. Because the global buffer may be used by multiple processes, this item must be checked if there are processes other than `pdpgbfon` that have accessed the global buffer.

[2] If the cause is an increase in the number of data items stored in the database, increase the global buffer size. If the cause is an increase in the number of storage pages due to disorganization of the database, execute the free page release utility (`pdreclaim`) or the database reorganization utility (`pdrorg`) to obtain the appropriate number of storage pages.

12.1.3 Execution environment

1. You can execute `pdpgbfon` only while HiRDB is running.
2. Execute the `pdpgbfon` command at the server machine containing the single server or the server machine where the system manager is located.
3. To execute `pdpgbfon`, set the `LANG` environment variable. To use character codes that are not supported by the OS in an environment in which `pdpgbfon` is executed, you must set the `PDLANG` environment variable. For details about `LANG` and `PDLANG`, see the *HiRDB Version 8 UAP Development Guide*.
4. You can execute `pdpgbfon` on an RDAREA as long as it is in open or command shutdown status. For details about whether or not `pdpgbfon` can be executed, see Appendix *C. RDAREA Status During Command Execution*.
5. When you will be executing `pdpgbfon`, allocate a sufficient number of global buffer sectors so that all data in the target table's data pages or index pages can be read. If a space shortage occurs in the global buffer, the first page read will be swept out of the global buffer.

12.1.4 Executor

- When data is read in units of schemas

  You must have the `DBA` privilege or you must be the owner of the schema.
- Other

  You must have the `DBA` privilege or the `SELECT` privilege for the table.

If you have the `DBA` privilege, you can also process the audit trail table. When the Directory Server linkage facility is used, the indicated privilege must be assigned to the executor's specified authorization identifier or role.

12.2 Examples

This section presents five examples of `pdpgbfon`.

(1) HiRDB/Single Server

Example 1

This example reads data pages of a table (`TBL1`) into the global buffer. The example assumes that the table and indexes are defined as follows:

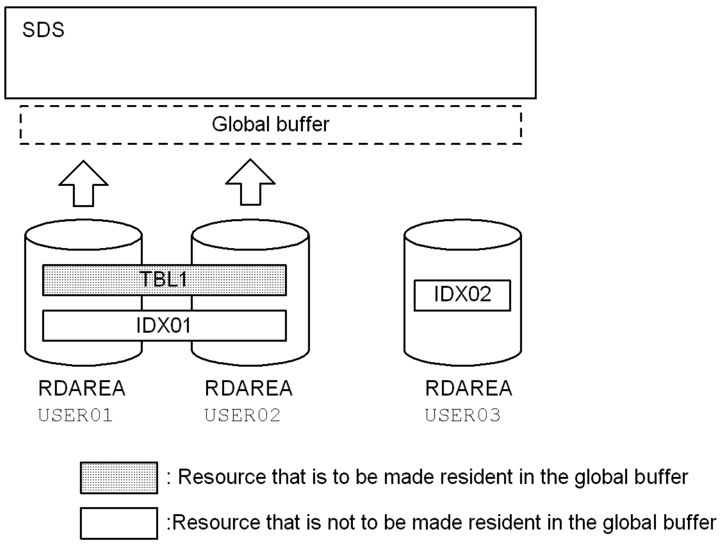
- Table definition

```
CREATE FIX TABLE TBL1(C1 INTEGER,C2 CHAR(10),C3 DEC(15))
    IN ((USER01) C1>0,(USER02))
```

- Index definitions

```
CREATE INDEX IDX01 ON TBL1(C1) IN ((USER01),(USER02))
CREATE INDEX IDX02 ON TBL1(C3) IN USER03
```

Overview


SDS
Global buffer
TBL1
IDX01
IDX02
RDAREA USER01
RDAREA USER02
RDAREA USER03
: Resource that is to be made resident in the global buffer
:Resource that is not to be made resident in the global buffer

- pdpgbfon command

```
pdpgbfon -k table -t TBL1
```

Explanation

`-k table`: Specifies that a table's data pages are to be read.

`-t TBL1`: Specifies the table to be read.

Example 2

This example reads index pages of a table (`TBL1`) into the global buffer. The example assumes that the table and indexes are defined as follows:

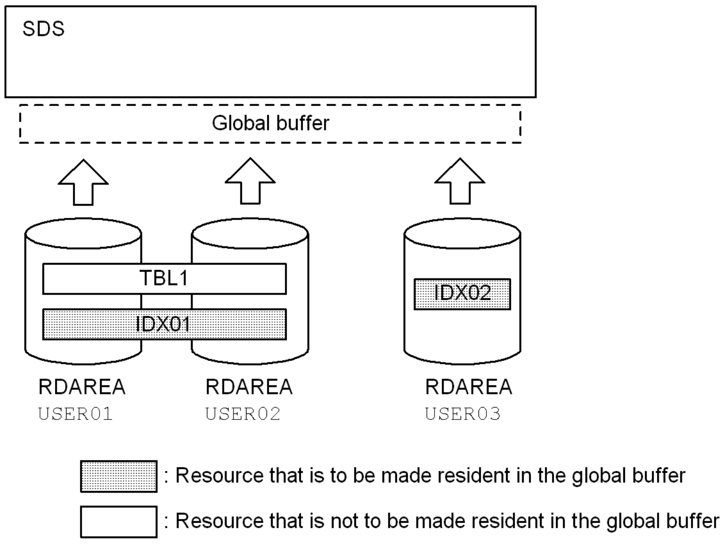
- Table definition

```
CREATE FIX TABLE TBL1(C1 INTEGER,C2 CHAR(10),C3 DEC(15))
    IN ((USER01) C1>0,(USER02))
```

- Index definitions

```
CREATE INDEX IDX01 ON TBL1(C1) IN ((USER01),(USER02))
CREATE INDEX IDX02 ON TBL1(C3) IN USER03
```

Overview


SDS
Global buffer
TBL1
IDX01
IDX02
RDAREA
USER01
RDAREA
USER02
RDAREA
USER03
: Resource that is to be made resident in the global buffer
: Resource that is not to be made resident in the global buffer

- pdpgbfon command

```
pdpgbfon -k index -t TBL1
```

Explanation

`-k index`: Specifies that a table's index pages are to be read.

`-t TBL1`: Specifies the table whose index pages are to be read.

(2) HiRDB/Parallel Server

Example 3

This example reads data pages of a table (`TBL1`) into the global buffer. The example assumes that the table and indexes are defined as follows:

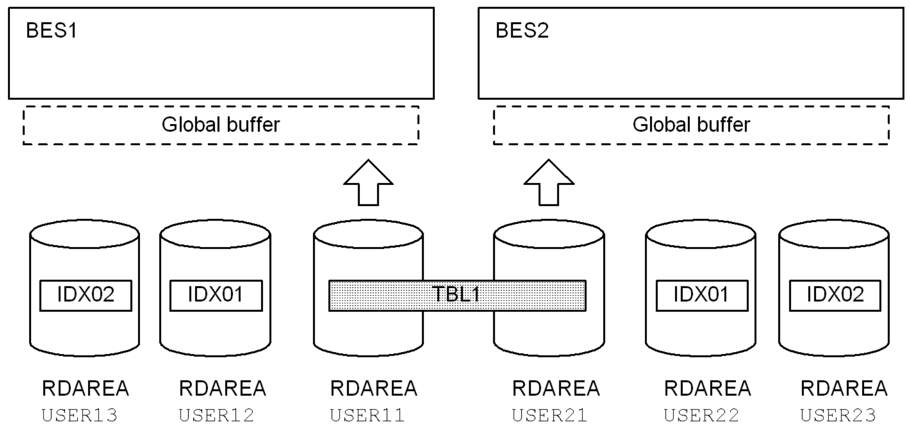
- Table definition

```
CREATE FIX TABLE TBL1(C1 INTEGER,C2 CHAR(10),C3 DEC(15))
    IN ((USER11) C1>0,(USER21))
```

- Index definitions

```
CREATE INDEX IDX01 ON TBL1(C1) IN ((USER12),(USER22))
CREATE INDEX IDX02 ON TBL1(C3) IN ((USER13),(USER23))
```

Overview



▨ : Resource that is to be made resident in the global buffer

☐ : Resource that is not to be made resident in the global buffer

- pdpgbfon command

```
pdpgbfon -k table -t TBL1
```

Explanation

`-k table`: Specifies that a table's data pages are to be read.

`-t TBL1`: Specifies the table to be read.

Example 4

This example reads index pages of a table (`TBL1`) into the global buffer in units of indexes. The example assumes that the table and indexes are defined as follows:

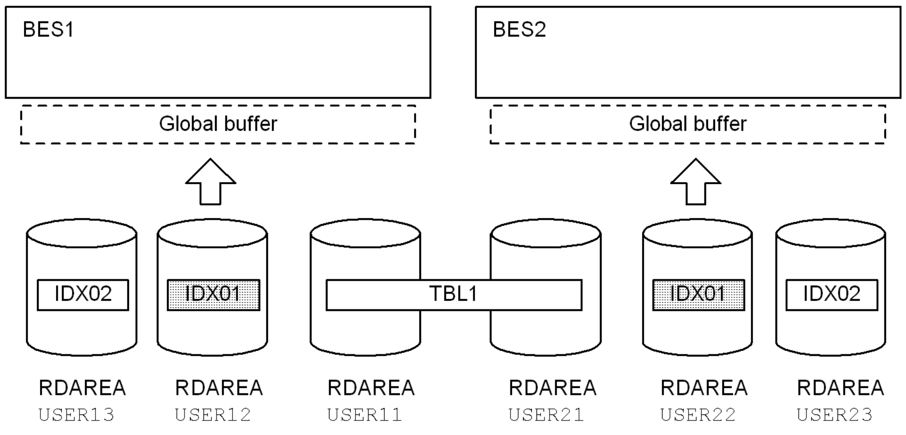
- Table definition

```
CREATE FIX TABLE TBL1(C1 INTEGER,C2 CHAR(10),C3 DEC(15))
    IN ((USER11) C1>0,(USER21))
```

- Index definitions

```
CREATE INDEX IDX01 ON TBL1(C1) IN ((USER12),(USER22))
CREATE INDEX IDX02 ON TBL1(C3) IN ((USER13),(USER23))
```

Overview



▨ : Resource that is to be made resident in the global buffer

▭ : Resource that is not to be made resident in the global buffer

- pdpgbfon command

```
pdpgbfon -k index -t TBL1 control_file
```

Explanation

`-k index`: Specifies that a table's index pages are to be read.

`-t TBL1`: Specifies the table whose index pages are to be read.

`control_file`: Specifies the control information file.

- Contents of the control information file (control_file)

```
idxname name=IDX01    [1]
```

Explanation

1. Specifies the name of the index that is to be read into the global buffer.

Example 5

This example reads index pages of a table (`TBL1`) into the global buffer in units of RDAREAs. The example assumes that the table and indexes are defined as follows:

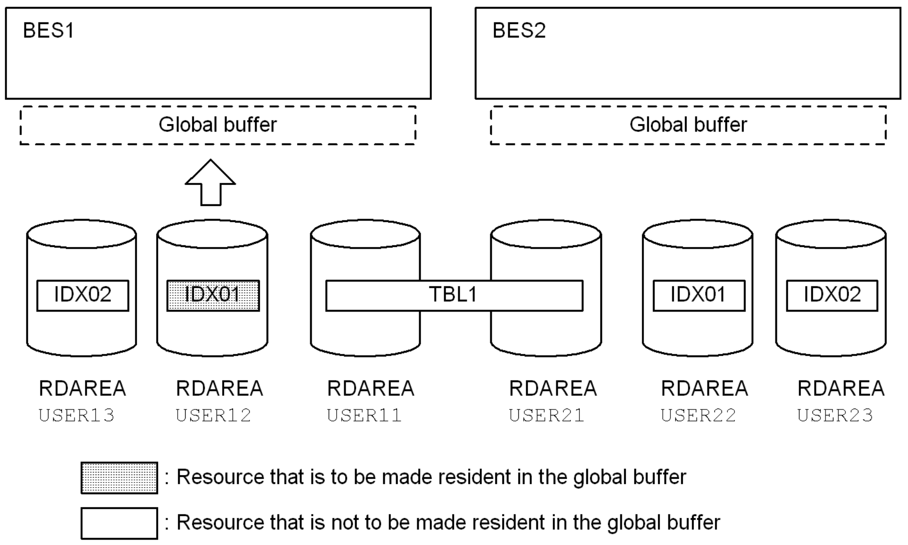
- Table definition

```
CREATE FIX TABLE TBL1(C1 INTEGER,C2 CHAR(10),C3 DEC(15))
    IN ((USER11) C1>0,(USER21))
```

- Index definitions

```
CREATE INDEX IDX01 ON TBL1(C1) IN ((USER12),(USER22))
CREATE INDEX IDX02 ON TBL1(C3) IN ((USER13),(USER23))
```

Overview



- pdpgbfon command

```
pdpgbfon -k index -t TBL1 control_file
```

Explanation

`-k index`: Specifies that a table's index pages are to be read.

`-t TBL1`: Specifies the table whose index pages are to be read.

`control_file`: Specifies the control information file.

- Contents of the control information file (control_file)

```
idxname name=IDX01 rdarea=USER12    [1]
```

Explanation

1. Specifies the names of the index and RDAREA that are to be read into the global buffer.

12.3 Command format

12.3.1 Format

This section describes the format of the `pdpgbfon` command. In the following table, each number corresponds to the number assigned to each option.

No.	Format
1	`pdpgbfon -k` *target-resource*
2	`-t {[`*authorization-identifier*`.]`*table-identifier*`|[`*authorization-identifier*`.]all}`
3	`[-r` *RDAREA-name*`]`
4	`[-u` *authorization-identifier*`]`
5	`[-X` *response-monitoring-time-for-server-to-server-communication*`]`
6	`[-q` *generation-number*`]`
7	`[-s` *server-name*`]`
8	`[-b]`
9	`[`*control-information-file-name*`]`

Note

If you specify *control-information-file-name*, be sure to specify it as the last option.

■ Relationships between options

Table 12-1 shows the relationships between options.

Table 12-1: Relationships between options (pdpgbfon)

Option	-k option		-t option	
	table	index	table-identifier	all
`-k` *target-resource*	—	—	M	M
`-t {[`*authorization-identifier*`.]`*table-identifier*`|[`*authorization-identifier*`.]all}`	M	M	—	—
`-r` *RDAREA-name*	Y	N	Y	N
`-u` *authorization-identifier*	Y	Y	Y	Y

Option	-k option		-t option	
	table	index	table-identifier	all
`-X` *response-monitoring-time-for-server-to-server-communication*	Y	Y	Y	Y
`-q` *generation-number*	Y	Y	Y	Y
`-s` *server-name*	Y	Y	Y	Y
`-b`	Y	N	Y	Y
control-information-file-name	Y	Y	Y	Y

Legend:

M: Specification is mandatory

Y: Can be specified

N: Cannot be specified

—: Not applicable

12.3.2 Options

(1) -k target-resource

Specifies the type of resource (data pages or index pages) that is to be read into the global buffer.

`table`:

Specifies that data pages are to be read.

`index`:

Specifies that index pages are to be read.

(2) -t {[authorization-identifier.]table-identifier|[authorization-identifier.]all}

~ <identifier>

Specifies the table name of the target resources.

When you omit the authorization identifier, the user name used to connect to HiRDB is assumed.

When `all` is specified, the utility reads all tables or indexes that belong to that authorization identifier (schema). In this case, the `idxname` statement cannot be specified.

You cannot specify a data dictionary table, view table, foreign table, or a table containing a user-defined abstract data-type column.

(3) -r RDAREA-name

~ <identifier>

Specifies the name of the RDAREA in the table specified in the `-t` option that is to be read into the global buffer.

This option is applicable to `-k table`. If you have specified `-k index`, use the `idxname` statement to specify this information.

A LOB RDAREA cannot be specified. A batch specification of RDAREA names cannot be used.

Criteria

For a table for a HiRDB/Parallel Server whose rows are partitioned among servers, the system performs parallel processing in units of RDAREAs using multiple server processes even when this option is not specified. On the other hand, in the case of a HiRDB/Single Server or a table for a HiRDB/Parallel Server whose rows are partitioned within a server, the system does not perform parallel processing because each RDAREA is processed by one server process. For the latter, you can use this option to perform parallel processing in units of RDAREAs using multiple server processes.

(4) -u authorization-identifier

Specifies the authorization identifier of the user who executes `pdpgbfon`.

For details about the authorization identifier, see *8.9.2(10) -u authorization-identifier*.

(5) -X response-monitoring-time-for-server-to-server-communication

~ <unsigned integer> ((1-65535)) <<300>>

If an error, such as a communication error, occurs at the server where the command was executed, the command may stop responding and the application may stop. To detect errors, `pdpgbfon` enables you to monitor a response time during communication for dictionary manipulation that is performed by the command.

You set in the `-X` option the response monitoring time during dictionary manipulation (in seconds). If the execution time during dictionary manipulation exceeds the value set in the `-X` option, `pdpgbfon` treats it as a dictionary access error and cancels processing with return code `8`.

Criteria

- If you want to detect an error in less time than 300 seconds in the event of a no-response from the server due to a communication error or unit down, specify a smaller value than 300 in the `-X` option.
- If the system switchover facility is used, the command may keep waiting for a response even though system switchover has been completed. In such a

case, you can terminate the command immediately by reducing the monitoring time.

- The specified monitoring time may result in a timeout if a response from the dictionary is delayed and if the utility's preprocessing is not completed within 300 seconds, which is the default value of the `-X` option. This can happen when many applications and utilities are executing concurrently. In such an environment, specify a value greater than 300 in the `-X` option.

(6) -q generation-number

~ <unsigned integer> ((0-10))

When the inner replica facility is used, specifies the generation number of the RDAREA to be read.

Specify the generation number as follows:

`0`: Original RDAREA is to be read.

`1` to `10`: Specified generation of the replica RDAREA that is to be read.

For details about the rules, see *11.3.2(9) -q generation-number*.

(7) -s server-name

~ <identifier>

Specifies the name of the back-end server that is to control execution of `pdpgbfon`.

This option is applicable to a HiRDB/Parallel Server only (for a HiRDB/Single Server, the option is ignored, if specified).

Criteria

Normally, there is no need to specify this option because the system automatically selects one of the back-end servers when the option is omitted.

Specify this option when the maximum number of concurrently activated server processes is exceeded because many instances of `pdpgbfon` are being executed concurrently.

(8) -b

Specifies that data stored in a branched `BINARY`-type column in a separate segment is to be read when a table data page is read and the table contains a `BINARY`-type column whose defined length is 256 bytes or greater.

If the table contains a `BINARY`-type column with a defined length of 256 bytes or greater and this option is omitted, the utility reads `BINARY`-type columns stored in the same segment as for a non-`BINARY`-type column, but not a branched `BINARY`-type column in a separate segment.

(9) control-information-file-name

~ <path name>

Specifies the name of the control information file that contains the control statements for `pdpgbfon`.

You can specify the following control statements in the control information file; for details about each control statement, see *12.3.3* and *12.3.4*:

- `idxname` statement
- `option` statement

12.3.3 idxname statement (specification of index information)

When index pages are to be read into the global buffer (`-k index`), the `idxname` statement specifies information about the target index.

Criteria

When `-k index` is specified, the utility processes all indexes defined for the table that is specified with the `-t` option. You specify the `idxname` statement to read only a particular index, server, or RDAREA.

Rules

1. Specify the `name` operand before the `server` and `rdarea` operands.
2. Enclose an identifier in double quotation marks (`"`) if it contains any spaces or lowercase letters. Lowercase letters are treated as uppercase letters, if not enclosed in double quotation marks.
3. Once you specify an `idxname` statement with the `rdarea` operand specified, you cannot specify another `idxname` statement specifying only the `name` operand or an `idxname` statement specifying the `server` operand.
4. You cannot specify more than one `idxname` statement specifying the same index identifier, except when the `rdarea` operand is specified.

(1) Format

```
idxname name={index-identifier|*}
         [{server=server-name[,server-name]...
              |rdarea=RDAREA-name[,RDAREA-name]...}]
```

(2) Explanation

(a) name={index-identifier|*}

index-identifier ~ <identifier>

Specifies the identifier of the index that is to be read into the global buffer.

There is no need to specify the authorization identifier, because the utility assumes the authorization identifier of the table specified with the `-t` option.

An asterisk (`*`) specifies that all indexes defined for the table are to be read.

The system reads all indexes for the table both when the `idxname` statement specifying `*` (without the `server` operand) is specified and when the `idxname` statement is omitted. If you specify the `idxname` statement specifying both `*` and the `server` operand, the utility reads only those indexes located at the back-end server specified in the `server` operand.

When `*` is specified, there can be only one `idxname` statement.

(b) server=server-name[,server-name]...

~ <identifier>

This operand is applicable to a HiRDB/Parallel Server only.

Specifies the name of the server (back-end server name) when only the portion of the index specified in the `name` operand that is located at the specified server is to be processed. When this operand is omitted, the utility assumes all servers that contain the index.

In the case of a shared index, the utility performs processing at the updatable back-end server and all reference-only back-end servers regardless of this operand's specification.

(c) rdarea=RDAREA-name[,RDAREA-name]...

~ <identifier>

Used to target specific index RDAREAs, this operand specifies the names of RDAREAs for the index that is specified in the `name` operand. When this option is omitted, the utility assumes all RDAREAs that store the index.

If `*` is specified in the `name` operand, this operand cannot be specified.

12.3.4 option statement (specification of optional information)

The `option` statement specifies `pdpgbfon`'s execution monitoring time.

Criteria

Specify the `option` statement to monitor the execution time of `pdpgbfon`.

(1) Format

```
option exectime=pdpgbfon-monitoring-time
```

(2) Explanation

(a) exectime=pdpgbfon-monitoring-time

~ <unsigned integer> ((0-35791394))

Specifies the `pdpgbfon` execution monitoring time in minutes.

Guideline for the specification value

The purpose of this operand is to detect a no-response error, not to monitor the execution time of a long transaction. Therefore, you should specify a sufficient time for processing the applicable table. For example, to monitor the execution time of a `pdpgbfon` that should terminate in 7-8 minutes, specify `exectime=20`, not `exectime=10`.

Rules

1. If you omit this operand or specify `0`, the system will not monitor the execution time.
2. This operand takes precedence over the `pd_utl_exec_time` operand in the system definition. Table 12-2 describes the relationship between the `exectime` operand and the `pd_utl_exec_time` operand in the system definition.

Table 12-2: Relationship between the exectime operand and the pd_utl_exec_time operand in the system definition (pdpgbfon)

exectime operand value	**pd_utl_exec_time operand value in the system definition**	
	Omitted or 0	**A**
Omitted	—	Monitoring with value `A`
`0`	—	—
`B`	Monitoring with value `B`	Monitoring with value `B`

Legend:

— : Execution time is not monitored.

3. If `pdpgbfon` processing does not terminate within the specified time, terminate the utility process forcibly and collect troubleshooting information to determine the cause of the no-response error. Note that this statement monitors the execution time of the single server (or back-end server) that actually issues access requests to the global buffer and the utility process (`0mrorg`). If an error occurs in the command process (`pdpgbfon` or `pdrorg`), the process cannot be terminated forcibly. For details about the error information to be collected, see the `pd_utl_exec_time` operand in the manual *HiRDB Version 8 System Definition*.

12.4 pdpgbfon processing results

When `pdpgbfon` terminates normally, its processing results are displayed at the standard output where the command was executed.

(1) When data pages are read into the global buffer

```
pdpgbfon VV-RR     *** BUFFER INFORMATION ***     YYYY-MM-DD
hh:mm:ss [1]

TABLE NAME : USERA.TBL1 [2]
 [3]  [4]              [5]            [6]           [7]
No.  SERVER          RFGET          READ         RFFLS
  1  bes1            1,275           596             0
  2  bes2            1,271           594             0
  3  bes3            1,273           596             0

 8365 20:03:54 un01         KFPL00738-I Pdpgbfon terminated,
return code=0
```

Explanation

1. Header for the processing results

   VV-RR: Version and revision numbers

   YYYY-MM-DD hh:mm:ss: Date and time execution of `pdpgbfon` started
2. Name of the resource that was processed
3. Serially assigned numbers
4. Names of the server
5. Reference `GET` counts*

   Each is the number of times `pdpgbfon` referenced the global buffer.

   This count is higher than the number of corresponding resource pages used because it includes the management (directory) page reference count. If a referenced page has already been read in the global buffer, no real READ occurs.
6. Real `READ` counts*

   Each is the number of input requests issued to the database by `pdpgbfon` because the target page was not found in the global buffer.
7. Reference buffer flush counts*

Each is the number of times an unupdated page was flushed out of the global buffer by the LRU method because a space shortage occurred in the global buffer when `pdpgbfon` read a page into the global buffer.

If this value is not `0` when `pdpgbfon` is executed while the global buffer contains no other resource page information (such as immediately after HiRDB startup), the global buffer is not large enough to read all pages.

* The maximum count that can be displayed is 4,294,967,295. If this value is exceeded, an overflow occurs.

(2) When index pages are read into the global buffer

```
pdpgbfon VV-RR      *** BUFFER INFORMATION ***      YYYY-MM-DD
HH:MM:SS [1]

INDEX NAME : USERA.(PRIMARY0000131193)[2]
 [3]  [4]              [5]            [6]            [7]
No.  SERVER           RFGET          READ          RFFLS
  1  bes1                75            69              0
  2  bes2                75            69              0
  3  bes3                75            69              0

INDEX NAME : USERA.IDX01
No.  SERVER           RFGET          READ          RFFLS
  1  bes1               525           478              0
  2  bes2               523           476              0
  3  bes3               524           477              0

 8373 20:04:04 un01         KFPL00738-I Pdpgbfon terminated,
return code=0
```

Explanation

See the explanation in *(1) When data pages are read into the global buffer*. Even when an index-dedicated global buffer is allocated, the utility uses the global buffer allocated to the RDAREA (if it is not specified, the global buffer specified in the `-o` option) to access the management (directory) page of the RDAREA that stores the corresponding index. Therefore, if a space shortage occurs in the global buffer allocated to the RDAREA, the reference buffer flush count is displayed even when there is sufficient space in the index-dedicated global buffer.

12.5 Notes

1. The maximum number of `pdpgbfon` commands that can be executed concurrently is the same as for the `pdrorg` command because `pdpgbfon` executes `pdrorg` internally. Specifically, the maximum number of `pdpgbfon` commands that can be executed concurrently equals the maximum number of `pdrorg` commands that can be executed concurrently minus the number of `pdrorg` commands that are currently executing.
2. `pdpgbfon` cannot process LOB RDAREAs. This means that `pdpgbfon` cannot process any LOB RDAREA containing an abstract data type or plug-in index.
3. To terminate `pdpgbfon` forcibly, use the `pdcancel` command. Other commands such as the OS's `kill` command cannot terminate `pdpgbfon`.
4. When a shared table or shared index is read, the updatable back-end server and all reference-only back-end servers are processed by `pdpgbfon`.
5. Executing `pdpgbfon` on an RDAREA that has the `SCHEDULE` attribute and that is not open serves no purposes (because the RDAREA will be closed after the processing).
6. For a HiRDB/Parallel Server, if a transmission message from the back-end server to `pdreclaim` is delayed for 1 second or more due to traffic on the communication line, the corresponding back-end server's processing result may not be displayed at the standard output. In such a case, you can check the `KFPL00714-I` message that is output to the message log file or `syslogfile` to determine the processing result.
7. If the target table contains a user-defined column of an abstract data type, `pdpgbfon` cannot be executed.
8. Table 12-3 shows the `pdpgbfon` return codes.

Table 12-3: Return codes of pdpgbfon

Return code	Description	Action
`0`	Global buffer read operation was completed.	None

Return code	Description	Action
`4`	Buffer miss occurred (some pages were not read), because there were not enough buffer sectors when data was read into the global buffer.	If necessary, increase the number of global buffer sectors. If a single global buffer is shared among multiple resources, a page of another resource may be read into the global buffer by another job before or during execution of `pdpgbfon`. In this case, sweeping a page of the other resource out of the global buffer is also treated as a buffer miss; therefore, whether the buffer miss applies to the target resource's page or another resource's page cannot be determined. If you are using only the return code to determine whether or not all pages were read into the global buffer successfully, do not execute any other job until `pdpgbfon` has terminated.
`8`	`pdpgbfon` terminated abnormally (global buffer read operation was cancelled).	The table and index status is guaranteed and can be accessed from UAPs even after abnormal termination. To re-execute `pdpgbfon`, see the displayed error message and eliminate the cause of the error.
`12`	`pdrorg` terminated abnormally (`pdrorg` that runs as an extension of `pdpgbfon` terminated abnormally).	The table and index status is guaranteed and can be accessed from UAPs even after abnormal termination. To re-execute `pdpgbfon`, see the displayed error message, eliminate the cause of the error, then re-execute `pdpgbfon`.

9. Do not execute a definition SQL statement on a table or index that is being processed by `pdpgbfon`. If a definition SQL statement is so executed, `pdpgbfon` terminates abnormally. If `pdpgbfon` is executing in units of schemas, it processes all tables and indexes that belong to the corresponding schema.
10. If you selected `utf-8` as the character encoding in the `pdsetup` command, you can use a file with a BOM as the control statements file for `pdpgbfon`. Note that even when a file with a BOM is used as the control statements file, the BOM is skipped.

Chapter

13. Integrity Check Utility (pdconstck)

This chapter explains the integrity check utility (`pdconstck`) that performs integrity checking and manipulates the check pending status on tables for which referential constraints or check constraints have been defined.

13.1 Overview

The integrity check utility (`pdconstck`) performs integrity checking and manipulates (sets or releases) the check pending status on tables for which referential constraints or check constraints have been defined.

(1) Prerequisites

(a) Operating conditions

Table 13-1 describes the prerequisites for executing `pdconstck`.

Table 13-1: Prerequisites for executing pdconstck

Item	Prerequisites
Execution privilege	A user with the `DBA` privilege or the subject table's owner can execute the utility.
RDAREA status	For details about the relationship between `pdconstck` and the status of an RDAREA to be processed, see *C. RDAREA Status During Command Execution*.
Concurrent execution	Only one instance of `pdconstck` can be executed on the same table at the same time. If multiple instances of `pdconstck` are executed concurrently, deadlock may occur between the subject table's `TABLE` resource type (resource type 0002) and `DICT` resource type (resource type 3005). Multiple instances of `pdconstck` can be executed concurrently as long as there is no duplication among the target tables. However, when the check pending status is changed, the utility is placed in lock-release wait status.
Maximum number of utilities that can be executed concurrently	The maximum number of instances of `pdconstck`s that can be executed concurrently is equal to the value of the `pd_max_users` operand/2.
Number of resources	Because `pdconstck` locks the target table and a referenced table that is referenced by a foreign key, you must pay attention to the following values in the system definition: • Number of base tables that can be accessed concurrently In the `pd_max_access_tables` operand in the system definition, specify a value that is at least (number of foreign keys for the target table + 1). • Sum of the number of locked tables and the number of RDAREAs for which `UNTIL DISCONNECT` is specified per server In the `pd_lck_until_disconnect_cnt` operand in the system definition, specify a value that is at least the value of *X* in the following formula: *X* = (2 + number of foreign keys for the target table + number of RDAREAs that store the target table + total number of RDAREAs that store the table whose primary key is referenced by the foreign keys)

Item	Prerequisites
	• Lock pool size per server In the `pd_lck_pool_size` operand in the system definition, specify a value that is at least $\uparrow Y \div x \uparrow$. In a HiRDB/Single Server: $Y = 209 + A + f + (a + c + (d \times b)) \times e$ In a HiRDB/Parallel Server: ● Dictionary server $Y = 209 + A + (a + c + (d \times b)) \times e$ ● Front-end server $Y = 2 + f$ ● Back-end server $Y = (d \times b) \times e$ Legend: *a*: Number of table storage RDAREAs *b*: Number of index storage RDAREAs *c*: Number of LOB column storage RDAREAs *d*: Number of indexes *e*: Number of generations that have replica RDAREAs *f*: Number of plug-ins defined for the target table and the table whose primary key is referenced by foreign keys *x*: 6 for 32-bit mode HiRDB, 4 for 64-bit mode HiRDB *A*: MAX (number of columns in the table, number of indexes for the table, number of table storage RDAREAs, number of foreign keys for the table, number of check constraints for the table)
Change of check pending status	Executing `pdconstck` changes the check pending status, regardless of the specification of the `pd_check_pending` operand in the system definition.

Item	Prerequisites
Inner replica facility	If you use the inner replica facility, make sure that all RDAREAs associated with the target table (table storage RDAREAs, LOB column storage RDAREAs, and index storage RDAREAs) have the same number of replica definitions. Also, make sure that all the replica RDAREAs are of the same generation. If the target table is a referencing table, `pdconstck` can be executed on the RDAREAs that are related to the table referenced by the referencing table even if its replica definition is different from that for the corresponding table. 1. Target table ● If all the replica RDAREAs related to the target table are generations that are in command shutdown and closed status, the table is not subject to processing. ● If the target table is partitioned among multiple servers and the local server contains no replica RDAREAs, the local server processes the current RDAREA. 2. Table (referenced table) that is referenced by the target table (referencing table) ● If all generations are subject to processing, and if the generation of the replica RDAREAs related to the referenced table that is referenced by the target table (referencing table) is different from the generation of the replica RDAREA that corresponds to the target table, integrity checking is performed on the current RDAREA. ● If an individual generation is subject to processing and the specified generation does not contain a replica RDAREA related to the referenced table that is referenced by the referencing table, integrity checking is performed on the current RDAREA. ● If the generation of the current RDAREA is subject to processing, integrity checking is performed on the current RDAREA for the referenced table that is referenced by the referencing table.

(b) Operating environment

Table 13-2 describes the operating environment for `pdconstck`.

Table 13-2: Operating environment for pdconstck

Item		Operating environment
HiRDB environment	Location of command execution	Execute `pdconstck` at the server machine that contains the single server, or where the system manager is located.
	Server status	When you execute `pdconstck`, the following server(s) must be running: In a HiRDB/Single Server: Single server In a HiRDB/Parallel Server: Dictionary server, front-end server, and back-end server (in the case of multiple front-end servers, at least one of them must be running)
	`PDDIR` environment variable	In the `PDDIR` environment variable, set the absolute path name of the HiRDB directory.
	`PDCONFPATH` environment variable	In the `PDCONFPATH` environment variable, set the absolute path name of the directory that stores the HiRDB system definition files.

Item		Operating environment
OS environment	`LANG` environment variable	In the `LANG` environment variable, set the value appropriate for the character encoding specified in the `pdsetup` command. For details about `LANG`, see the manual *HiRDB Version 8 UAP Development Guide*.
	File and directory specified in the `-o` option	Access privileges must have been granted to the user who executes `pdconstck`. If the `-o` option was omitted, access privileges must be granted in the same manner; otherwise, the system assumes that files are to be created in the `/tmp` directory.

(2) Authorization identifier during execution of pdconstck

When you execute `pdconstck`, you can specify the authorization identifier used to connect to HiRDB as well as the authorization identifiers for the target table and the constraint name. Table 13-3 lists the authorization identifier used to connect to HiRDB and the authorization identifiers for the target table and constraint name.

Table 13-3: Authorization identifier used to connect to HiRDB and authorization identifiers for the target table and constraint name

Specification condition				Connection with HiRDB		Auth identifier for the target table or constraint name
Auth identifier used to connect to HiRDB (-u option)	PDUSER environment variable		Auth identifier for target table or constraint name (-t or -c option)	Auth identifier	Password	
	Auth identifier	Password				
Specified	Specified	Specified	Specified	Authorization identifier specified in the `-u` option	Entered in response to a message*	Authorization identifier specified in the `-t` or `-c` option
			Not specified			Authorization identifier specified in the `-u` option
		Omitted	Specified			Authorization identifier specified in the `-t` or `-c` option
			Not specified			Authorization identifier specified in the `-u` option
	`PDUSER` environment variable omitted		Specified			Authorization identifier specified in the `-t` or `-c` option
			Not specified			Authorization identifier specified in the `-u` option

Specification condition				Connection with HiRDB		Auth identifier for the target table or constraint name
Auth identifier used to connect to HiRDB (-u option)	PDUSER environment variable		Auth identifier for target table or constraint name (-t or -c option)	Auth identifier	Password	
	Auth identifier	Password				
Not specified	Specified	Specified	Specified	Authorization identifier specified in the `PDUSER` environment variable	Password specified in the `PDUSER` environment variable	Authorization identifier specified in the `-t` or `-c` option
			Not specified			Authorization identifier specified in `PDUSER` environment variable
		Omitted	Specified		Omitted	Authorization identifier specified in the `-t` or `-c` option
			Not specified			Authorization identifier specified in `PDUSER` environment variable
	`PDUSER` environment variable omitted		Specified	User name (authorization identifier) in the login window	Entered in response to a message*	Authorization identifier specified in the `-t` or `-c` option
			Not specified			User name (authorization identifier) in the login window

Legend:

Auth: Authorization

* A message requesting entry of a password is displayed. Enter the appropriate password as the response to this message. If NULL is entered as the response, the system assumes that no password was set.

13.2 Functions of pdconstck

`pdconstck` has the following two functions:

Integrity check facility

Checks for constraint integrity errors in a table for which referential constraints or check constraints have been defined. This facility also releases or sets the table's check pending status based on whether or not a constraint violation is detected.

Facility for changing check pending status forcibly

Forcibly sets or releases a table's check pending status.

This section provides the details of these facilities.

13.2.1 Integrity check facility

The integrity check facility checks referential constraints or check constraints for integrity errors. If no row violates any constraints, the facility releases the check pending status.

Integrity checking on referential constraints

Based on the referential constraints, the facility checks to see if the foreign key of the target table has the same value as the primary key of the table referenced by the target table. If the checking result indicates that no row violates the constraints, the facility releases the check pending status. If there is a row that violates the constraints, the facility sets the check pending status. Note that the facility does not check rows whose foreign key value is the null value. Integrity checking on referential constraints may take time, depending on the size of the database. To determine the execution time, use the SQL statement presented in *(2)(a) How to perform integrity checking on referential constraints*.

Integrity checking on check constraints

The facility checks to see if row values are within the ranges of the check constraints in the check constraint definitions. If the checking results indicate that no row violates the constraints, the facility releases the check pending status. If any row violates a constraint, the facility sets the check pending status. Integrity checking on check constraints may take time, depending on the size of the database. To determine the execution time, use the SQL statement presented in *(3)(a) How to perform integrity checking on check constraints*.

(1) Execution unit of the integrity check facility

You can execute the integrity check facility in units of a table or a constraint, as explained below.

- By table

  All referential constraints and check constraints defined for the table are subject to processing.
- By constraint

  Only one constraint is subject to processing.

When the inner replica facility is used to perform processing by table, the integrity check facility can be executed in the following units:

- By all generations

  The original RDAREA and all generations of its replica RDAREAs are subject to processing.
- By generation

  Only one generation is subject to processing.
- By the current RDAREA's generation

  The current RDAREA's generation is subject to processing.

(a) By table

The facility checks all of the referential and check constraints defined for the table that have been placed in check pending status. You specify the `-t` option to perform integrity checking on a table.

Integrity checking by table is applied in the following cases:

- When a single execution of `pdconstck` is used both to check for violations in the referential and check constraints defined for the table that have been placed in check pending status, and to release their check pending status.
- When an RDAREA is to be re-initialized by `pdmod` to release tables and constraints from the RDAREA's check pending status, such as when the data dictionary table cannot be released from check pending status (for details, see *7.5.4 Table and index status after reinitialization*).

By all generations

For every referential constraint and check constraint that has been defined for the table and that has resulted in check pending status, integrity checking is performed on the original RDAREA and on each of its replica RDAREA generations that stores the table. The criteria are the same as for checking by table. You specify `-q all` to perform integrity checking on all generations.

By generation

For every referential constraint and check constraint that has been defined for the table and that has resulted in check pending status, integrity checking is performed only on the generation specified in the `-q` option. You specify the desired generation number in the `-q` option to perform integrity checking by generation.

Integrity checking by generation is applied when only the specific generation resulting in the check pending status is to be checked for violations, and when only its check pending status is to be released.

By the current RDAREA's generation

For every referential constraint and check constraint that has been defined for the table and that has resulted in check pending status, integrity checking is performed on the current RDAREA's generation. You omit the `-q` option to perform integrity checking by generation of the current RDAREA.

If a utility such as `pdload` is executed on the current RDAREA's generation and if the table results in check pending status, integrity checking by the current RDAREA's generation is applied in order to check only the current RDAREA's generation for violations, and then to release the check pending status.

(b) By constraint

In this case the facility checks only one constraint at a time for integrity errors, regardless of the check pending status. You specify the `-c` option to perform integrity checking by constraint. When the inner replica facility is used, integrity checking is performed on the original RDAREA that stores the table for which the constraint has been specified, as well as on all generations of its replica RDAREAs.

Integrity checking by constraint is applied in the following case:

- When an individual constraint is to be checked for a violation.

(2) Integrity checking on referential constraints

(a) How to perform integrity checking on referential constraints

To perform integrity checking for referential constraints, you use the SQL statement shown below for each constraint to be checked. When the inner replica facility is used, integrity checking is performed on the applicable constraint in a generation.

- SQL statement for integrity checking on a referential constraint

```
SELECT  DISTINCT "foreign-key-component-column-1", "foreign-key-component-column-2", ...,
"foreign-key-component-column-n"
FROM "referencing-table's-owner-name"."referencing-table's-table-identifier" RIC_REF_TBL
WHERE ("foreign-key-component-column-1" IS NOT NULL AND
      "foreign-key-component-column-2" IS NOT NULL AND ... "foreign-key-component-column-n" IS
NOT NULL)
AND NOT EXISTS (SELECT * FROM "referenced-table's-owner-name"."referenced-table's-table-identifier"
RIC_RFD_TBL
WHERE (RIC_RFD_TBL."primary-key-component-column-1",
RIC_RFD_TBL."primary-key-component-column-2",
        ..., RIC_RFD_TBL."primary-key-component-column-n") =
      (RIC_REF_TBL."foreign-key-component-column-1",
RIC_REF_TBL."foreign-key-component-column-2",
        ..., RIC_REF_TBL."foreign-key-component-column-n"))
ORDER BY  "foreign-key-component-column-1" ASC|DESC, "foreign-key-component-column-2" ASC|DESC,
          ..., "foreign-key-component-column-n" ASC|DESC
WITHOUT LOCK NOWAIT
```

Note:

Because this SQL statement must create a work table, a work table file for storing the work table is required in the single server or back-end server. For details about how to estimate the size of a work table file, see the manual *HiRDB Version 8 Installation and Design Guide*.

(b) Checking whether or not integrity has been maintained

Table 13-4 shows the relationship between the execution result (`SQLCODE`) of the SQL shown in (a) and whether or not integrity has been maintained.

Table 13-4: Relationship between SQL's execution result and whether or not integrity has been maintained (referential constraint)

SQLCODE (number of selection rows)	Whether or not integrity has been maintained
`100` (`0`)	Yes
`0` (`1` or greater)	No
Other	Execution error

(c) Output of key values resulting in constraint violations

After execution of the SQL statement described in (a), the key values resulting in constraint violations are output to the process results file. The key values resulting in constraint violations are output after duplicates have been eliminated.

(d) Cancellation of integrity checking because of the maximum number of key value violations that can be output

If the number of row data items resulting in constraint violations exceeds the maximum output count for violation key values, the facility terminates integrity checking on that constraint (generation). You use the `-w` option to change the maximum number of violation key values that can be output.

(3) Integrity checking on check constraints

(a) How to perform integrity checking on check constraints

To perform integrity checking for check constraints, you use the SQL statement shown below for each constraint to be checked. When the inner replica facility is used, integrity checking is performed on the applicable constraint in a generation.

- SQL statement executed for integrity checking on a check constraint

```
SELECT DISTINCT* "check-constraint-target-column-1", "check-constraint-target-column-2", ...,
"check-constraint-target-column-n"
FROM
"name-of-owner-of-table-for-which-check-constraint-was-defined"."table-identifier-of-table-for-which-check-c
onstraint-was-defined"
WHERE NOT(check-constraint-search-condition) WITHOUT LOCK NOWAIT
```

Note:

Because this SQL statement must create a work table, a work table file for storing the work table is required in the single server or back-end server. For details about how to estimate the size of a work table file, see the manual *HiRDB Version 8 Installation and Design Guide*.

* `DISTINCT` must be omitted when the column definition specified in the search condition is the `BLOB` or `BINARY` type. In this case, no work table is created.

(b) Checking whether or not integrity has been maintained

Table 13-5 shows the relationship between the execution result (`SQLCODE`) of the SQL shown in (a) and whether or not integrity has been maintained.

Table 13-5: Relationship between SQL's execution result and whether or not integrity has been maintained (check constraint)

SQLCODE (number of selection rows)	Whether or not integrity has been maintained
`100` (`0`)	Yes
`0` (`1` or more)	No
Other	Execution error

(c) Output of key values resulting in constraint violation

After execution of the SQL statement described in (a), the key values resulting in constraint violations are output to the process results file. If the data types of the columns specified in the search conditions do not include `BLOB` or `BINARY`, the key values resulting in constraint violations are output after duplicates have been eliminated. If the columns include any that are of the `BLOB` or `BINARY` type, duplicates of key values resulting in constraint violations are not eliminated.

(d) Cancellation of integrity checking because of the maximum number of key value violations that can be output

If the number of row data items resulting in constraint violations exceeds the maximum output count for violation key values, the facility terminates integrity checking on that constraint (generation). You use the `-w` option to change the maximum number of violation key values that can be output.

(4) Changing the check pending status based on the results of integrity checking

`pdconstck` sets a table, constraint, or RDAREA in check pending status or non-check pending status on the basis of the results of integrity checking for each constraint (and when the inner replica facility is used, for each generation for each constraint). The change in the check pending status for an RDAREA depends on the unit of execution of `pdconstck`.

This subsection describes the change in check pending status for a constraint, table, and RDAREA. A change in check pending status for a table, constraint, or RDAREA occurs separately for check constraints and referential constraints.

Change in the check pending status for a constraint

The check pending status for constraints changes for each constraint (each generation) based on the results of integrity checking. Table 13-6 shows the change in the check pending status for a constraint.

Table 13-6: Change in the check pending status for a constraint

Inner replica facility	Result of integrity checking on a constraint*	Constraint check pending status (setting)
Not used	Integrity is maintained	Non-check pending status (NULL value)
	Integrity is not maintained	Check pending status (`'C'`)
Used	Integrity is maintained in all generations	Non-check pending status (NULL value)
	Integrity is maintained in some of the generations	Check pending status (`'C'`)

* The facility assumes that integrity is maintained for a constraint (generation) whose integrity was not checked.

Change in the check pending status for a table

The check pending status for a table changes based on each constraint's check pending status. Table 13-7 shows the change in the check pending status for a table.

Table 13-7: Change in the check pending status for a table

Constraint check pending status	Table check pending status (setting)
All constraints are in non-check pending status	Non-check pending status (NULL value)
At least one constraint is in check pending status (even if the others are in non-check pending status)	Check pending status (`'C'`)

Change in the check pending status for an RDAREA

The check pending status for an RDAREA changes for each constraint (each generation) based on the results of integrity checking. Table 13-8 shows the change in the check pending status for an RDAREA.

Table 13-8: Change in the check pending status for an RDAREA

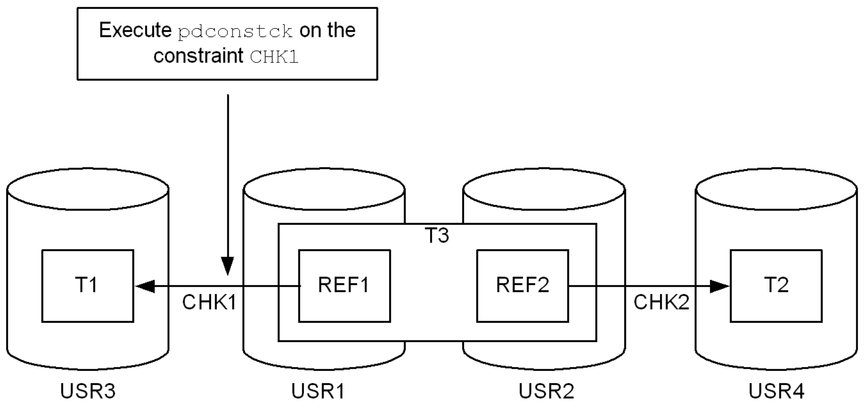
Result of integrity checking	RDAREA check pending status (setting)
Integrity is maintained for all constraints	Non-check pending status (NULL value)
Integrity is not maintained for at least one constraint (even if it is maintained for the others)	Check pending status (`'C'`)

(a) Changes in the check pending status when integrity checking is performed by table

When integrity checking is performed by table, the check pending status of the table, constraints, or RDAREAs changes based on the results of integrity checking for each constraint.

Figure 13-1 shows an example of executing integrity checking by table on a table for which referential constraints have been defined. Table 13-9 shows the changes in the check pending status in this example.

Figure 13-1: Example of integrity checking by table



Legend:

`T1`, `T2`: Tables in which the primary key has been defined

`T3`: Table in which the foreign keys (`REF1` and `REF2`) have been defined

`CHK1`, `CHK2`: Referential constraints

`USR1` to `USR4`: User RDAREAs storing the table

`REF1`, `REF2`: Foreign keys that reference another table

Table 13-9: Changes in the check pending status by table (example)

Result of integrity checking		**Check pending status**				
Integrity of REF1	**Integrity of REF2**	**Table check pending status**	**Constraint check pending status**		**Table storage RDAREA check pending status**	
			REF1	**REF2**	**USR1**	**USR2**
Maintained	Maintained	N	N	N	N	N
	Lost	P	N	P	P	P
Lost	Maintained	P	P	N	P	P
	Lost	P	P	P	P	P

Legend:

P: Check pending status

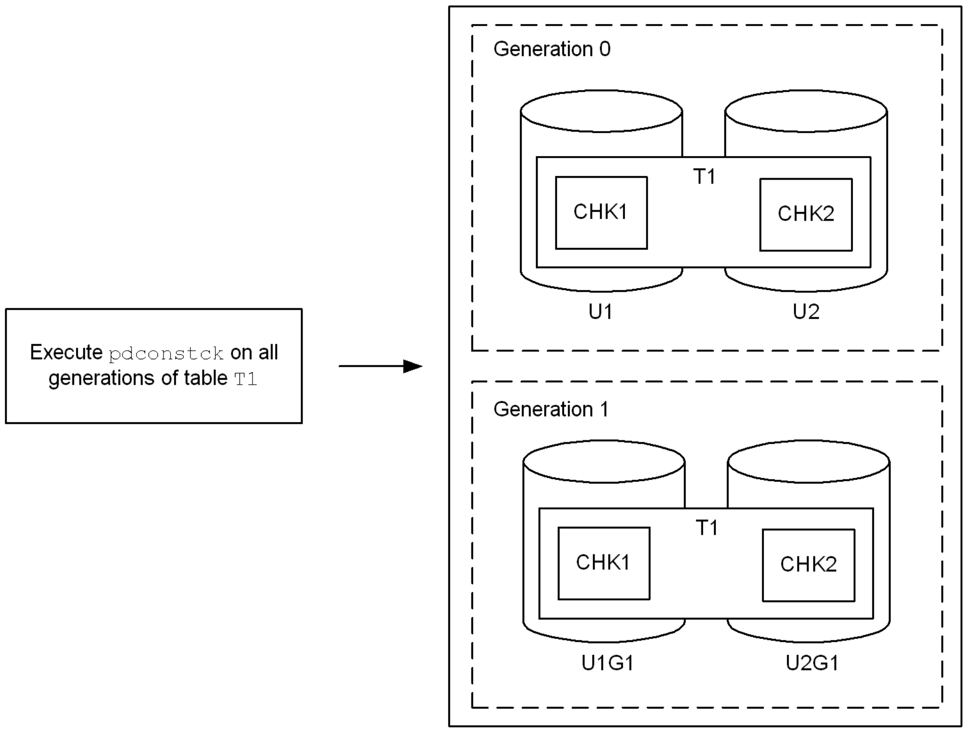
N: Non-check pending status

(b) Changes in the check pending status when integrity checking is performed on all generations

If integrity checking is performed on all generations when the inner replica facility has been used, the check pending status of the table, constraints, or RDAREAs changes based on the results of integrity checking for each constraint (or each generation).

Figure 13-2 shows an example of performing integrity checking on all generations when the inner replica facility has been used to create one generation of replica RDAREAs containing a table for which check constraints have been defined. Table 13-10 shows the changes in the check pending status in this example.

Figure 13-2: Example of integrity checking by all generations



Legend:

`U1`, `U2`: Original RDAREAs

`U1G1`, `U2G1`: Replica RDAREAs

`CHK1`, `CHK2`: Check constraints for table `T1`

Table 13-10: Changes in the check pending status by all generations (example)

Result of integrity checking				Check pending status						
Integrity of CHK1		Integrity of CHK2		Table check pend status	Constraint check pending status		RDAREA check pending status			
GEN 0	GEN 1	GEN 0	GEN 1		CHK1	CHK2	U1	U2	U1G1	U2G1
Mntd	Mntd	Mntd	Mntd	N	N	N	N	N	N	N
Mntd	Mntd	Mntd	Lost	P	N	P	N	N	P	P
Mntd	Mntd	Lost	Mntd	P	N	P	P	P	N	N
Mntd	Mntd	Lost	Lost	P	N	P	P	P	P	P
Mntd	Lost	Mntd	Mntd	P	P	N	N	N	P	P
Mntd	Lost	Mntd	Lost	P	P	P	N	N	P	P
Mntd	Lost	Lost	Mntd	P	P	P	P	P	P	P
Mntd	Lost	Lost	Lost	P	P	P	P	P	P	P
Lost	Mntd	Mntd	Mntd	P	P	N	P	P	N	N
Lost	Mntd	Mntd	Lost	P	P	P	P	P	P	P
Lost	Mntd	Lost	Mntd	P	P	P	P	P	N	N
Lost	Mntd	Lost	Lost	P	P	P	P	P	P	P
Lost	Lost	Mntd	Mntd	P	P	N	P	P	P	P
Lost	Lost	Mntd	Lost	P	P	P	P	P	P	P
Lost	Lost	Lost	Mntd	P	P	P	P	P	P	P
Lost	Lost	Lost	Lost	P	P	P	P	P	P	P

Legend:

GEN: Generation

Table check pend status: Table check pending status

Mntd: Maintained

P: Check pending status

N: Non-check pending status

(c) Changes in the check pending status when integrity checking is performed by generation

When the inner replica facility is used and integrity checking is performed by generation, the check pending status of the table, constraints, or RDAREAs changes based on the results of integrity checking on the specified generation for each constraint.

Table 13-11 shows the changes in the check pending status of a constraint and RDAREA based on the results of integrity checking by generation.

Table 13-11: Changes in the check pending status of a constraint and RDAREA based on the results of integrity checking by generation

Execution result			**Constraint check pending status**	**RDAREA check pending status**
Result of checking on the corresponding generation		**Check pending status in other generations of the RDAREA**		
Integrity is maintained in all constraints		All non-check pending status	All constraints are placed in non-check pending status.	The corresponding generation is placed in non-check pending status.
		Other	No change	The corresponding generation is placed in non-check pending status.
Other	Constraint whose integrity is maintained	—	No change	The corresponding generation is placed in check pending status.
	Constraint whose integrity is lost	—	Placed in check pending status.	The corresponding generation is placed in check pending status.

Legend:

— : Not applicable

Table 13-12 shows the changes in the check pending status when integrity checking is performed by generation on Generation 1 based on the example shown in Figure 13-2.

Table 13-12: Changes in the check pending status by generation (example)

Result of integrity checking		RDAREA check pending status	Check pending status						
			Table check pending status	Constraint check pending status		RDAREA check pending status			
Integrity of CHK1	**Integrity of CHK2**	**GEN 0**		**CHK1**	**CHK2**	**U1G1**	**U2G1**	**U1**	**U2**
Mntd	Mntd	Non-check pending status	N	N	N	N	N	(N)	(N)
Mntd	Mntd	Check pending status	(P)	(P)	(P)	N	N	(P)	(P)
Mntd	Lost	Non-check pending status	P	(P)	P	P	P	(N)	(N)
Mntd	Lost	Check pending status	P	(P)	P	P	P	(P)	(P)
Lost	Mntd	Non-check pending status	P	P	(P)	P	P	(N)	(N)
Lost	Mntd	Check pending status	P	P	(P)	P	P	(P)	(P)
Lost	Lost	Non-check pending status	P	P	P	P	P	(N)	(N)
Lost	Lost	Check pending status	P	P	P	P	P	(P)	(P)

Legend:

GEN: Generation

Mntd: Maintained

P: Check pending status

N: Non-check pending status

(P): No change (remains in check pending status)

(N): No change (remains in non-check pending status)

(d) Changes in the check pending status when integrity checking is performed by the current RDAREA's generation

When the inner replica facility is used and integrity checking is performed by the current RDAREA's generation, the check pending status of the table, constraints, or the specified generation of RDAREA changes based on the results of integrity checking on the specified generation for each constraint.

The changes in the check pending status when integrity checking is performed by the current RDAREA's generation are the same as in *(c) Changes in the check pending status when integrity checking is performed by generation*.

(e) Changes in the check pending status when integrity checking is performed by constraint

When integrity checking is performed by constraint, the check pending status of the table, the specified constraint, or the RDAREA changes based on the results of integrity checking on the specified constraint. The following describes the changes in the check pending status.

Changes in the check pending status (when the inner replica facility is not used)

When the inner replica facility is not used, the constraint's check pending status changes based on the results of integrity checking on the specified constraint.

Table 13-13 shows the changes in the check pending status when integrity checking is performed by constraint on constraint `CHK1` based on the example shown in Figure 13-1.

Table 13-13: Changes in the check pending status by constraint (example)

Result of integrity checking	**Constraint check pending status**	**Check pending status**				
Integrity of CHK1	**CHK2**	**Table check pending status**	**Constraint check pending status**		**RDAREA check pending status**	
			CHK1	**CHK2**	**U1**	**U2**
Maintained	Non-check pending status	N	N	—	N	N
Maintained	Check pending status	P	N	—	P	P
Lost	Non-check pending status	P	P	—	P	P

Result of integrity checking	**Constraint check pending status**	**Check pending status**				
Integrity of CHK1	**CHK2**	**Table check pending status**	**Constraint check pending status**		**RDAREA check pending status**	
			CHK1	**CHK2**	**U1**	**U2**
Lost	Check pending status	P	P	—	P	P

Legend:

P: Check pending status

N: Non-check pending status

— : The status remains unchanged.

Changes in the check pending status (when the inner replica facility is used)

When the inner replica facility is used, the check pending status of a constraint, each generation of the RDAREA, and the table change based on the results of integrity checking performed on the specified constraint in each generation.

Table 13-14 shows the changes in the check pending status when integrity checking is performed by constraint on constraint `CHK1` based on the example shown in Figure 13-2.

Table 13-14: Change in the check pending status by generation (example)

Result of integrity checking		**Other check pending status**	**Check pending status**						
			Table check pending status	**Constraint check pending status**		**RDAREA check pending status**			
Integrity of CHK1 generation 0	**Integrity of CHK1 generation 1**	**CHK2**		**CHK1**	**CHK2**	**U1**	**U2**	**U1G1**	**U2G1**
Maintained	Maintained	Non-check pending status	N	N	—	N	N	N	N
Maintained	Maintained	Check pending status	P	N	—	P	P	P	P

Result of integrity checking		Other check pending status	Check pending status						
			Table check pending status	Constraint check pending status		RDAREA check pending status			
Integrity of CHK1 generation 0	Integrity of CHK1 generation 1	CHK2		CHK1	CHK2	U1	U2	U1G1	U2G1
Maintained	Lost	Non-check pending status	P	P	—	P	P	P	P
Maintained	Lost	Check pending status	P	P	—	P	P	P	P
Lost	Maintained	Non-check pending status	P	P	—	P	P	P	P
Lost	Maintained	Check pending status	P	P	—	P	P	P	P
Lost	Lost	Non-check pending status	P	P	—	P	P	P	P
Lost	Lost	Check pending status	P	P	—	P	P	P	P

Legend:

P: Check pending status

N: Non-check pending status

— : The status remains unchanged.

(5) Order of integrity checking and whether or not integrity checking is executed

(a) By table

Order of integrity checking

When integrity checking is performed by table, the order in which it is performed is as follows:

Order between referential and check constraints

If referential constraints and check constraints are both defined for the table specified in the `-t` option, the referential constraints (foreign keys) are checked first, and then the check constraints are checked.

Order within the same type of constraints

If multiple constraints of the same type are defined for the table specified in the `-t` option, integrity is checked in the alphanumeric order of the constraint names.

Table 13-15 shows an example of the order in which integrity checking is performed by table.

Table 13-15: Example of order in which integrity checking is performed by table

Constraint for table		**Order of integrity checking**			
Referential constraint (foreign key)	**Check constraint**	**Referential constraint (foreign key)**		**Check constraint**	
		F1	**F2**	**C1**	**C2**
Maintained	Maintained	1.	2.	3.	4.
	Lost	1.	2.	—	—
Lost	Maintained	—	—	1.	2.
	Lost	—	—	—	—

Legend:

`F1`, `F2`: Constraint names for referential constraints (foreign key)

`C1`, `C2`: Constraint names for check constraints

1. to 4.: Order in which constraint checking is performed

—: Not applicable

Order of integrity checking by generation and whether or not integrity checking is executed by generation when the inner replica facility is used

Integrity checking by table (by all generations) is performed on the original RDAREAs (generation number 0) and then on their replicas for each constraint in ascending order of the generation numbers (starting at 1).

Whether or not integrity checking is executed

Integrity checking by table is performed only on constraints resulting in check pending status. When the inner replica facility is used, integrity checking is performed only on the generations of a constraint that have resulted in check pending status. Table 13-16 shows the conditions for performing integrity checking and whether or not integrity checking is executed.

Table 13-16: Conditions for performing integrity checking and whether or not integrity checking is executed

Check pending status			Whether or not integrity checking is executed	Remarks
Table check pending status	**Constraint check pending status**	**Table storage RDAREA check pending status**[1]		
Check pending status (`'C'`)	Check pending status (`'C'`)	Check pending status	Yes	`pdconstck` determines that the status is normal (check pending status).
		Non-check pending status	Yes	`pdconstck` checks integrity because the data dictionary table is in check pending status although the RDAREA is in non-check pending status.
	Non-check pending status (NULL value)[2]	Check pending status	No	`pdconstck` determines that the status is normal (other constraints are in check pending status).
		Non-check pending status	Yes	`pdconstck` checks integrity because the data dictionary table is in check pending status although the RDAREA is in non-check pending status.
Non-check pending status (NULL value)	Check pending status (`'C'`)	Check pending status	No	`pdconstck` terminates with an error because the data dictionary table is invalid.
		Non-check pending status	No	
	Non-check pending status (NULL value)[2]	Check pending status	Yes	`pdconstck` checks integrity because the RDAREA is in check pending status.
		Non-check pending status	No	`pdconstck` determines that the status is normal (non-check pending status).

1

A partitioned table is placed in check pending status or non-check pending status depending on the table information in each RDAREA that stores the table. Table 13-17 shows the check pending status based on the table information in each table storage RDAREA.

When the inner replica facility is used, the status (check pending status or non-check pending status) depends on the generation. Table 13-18 shows the check pending status based on the table information in each table storage RDAREA in the generation.

2

If consistency is lost between the table's check pending status and the constraint's check pending status, `pdconstck` terminates with an error. Table 13-19 shows the `pdconstck` processing when the data dictionary table is in check pending status.

Table 13-17: Check pending status based on the table information in each table storage RDAREA

Condition	Table information in table storage RDAREA
All storage RDAREAs are in check pending status	Check pending status
Some storage RDAREAs are in check pending status and some are in non-check pending status	Check pending status
All storage RDAREAs are in non-check pending status	Non-check pending status

Table 13-18: Check pending status based on the table information in each table storage RDAREA in the generation

Condition	Table information in table storage RDAREA
All storage RDAREAs subject to processing in the generation are in check pending status	Check pending status
Storage RDAREA subject to processing in the generation is in non-check pending status	Check pending status
All storage RDAREAs subject to processing in the generation are in non-check pending status	Non-check pending status

Table 13-19: pdconstck processing when the data dictionary table is in check pending status

Check pending status		pdconstck processing
Table check pending status	**Constraint check pending status**	
Check pending status	Some constraints were placed in check pending status	Resumes processing
	All constraints are in non-check pending status	Terminates with an error
Non-check pending status	Some constraints were placed in check pending status	Terminates with an error
	All constraints are in non-check pending status	Resumes processing

(b) By constraint

Order of integrity checking

When integrity checking is performed by constraint, there is no specific order in which integrity checking is performed, because only constraints are checked. However, if the inner replica facility is used, integrity is checked in the following order:

Checking order by generation when the inner replica facility is used

Integrity checking by constraint is performed on the original RDAREAs (generation number 0) and then on their replicas in ascending order of the generation numbers (starting at 1).

Whether or not integrity checking is executed

Integrity checking by constraint is performed in the above order, regardless of the table's check pending status.

13.2.2 Facility for changing check pending status forcibly

(1) Overview of function

The facility for changing check pending status forcibly changes the check pending status; this facility does not check the integrity of referential constraints or check constraints. This facility provides two functions, *forced setting of check pending status* and *forced release of check pending status*.

Forced setting of check pending status

This function forces a table into check pending status. You specify `-k set` to use this function.

Forced setting of check pending status is used in the following cases:

- The `pd_check_pending` operand value in the system definition was changed from `NOUSE` to `USE` and the integrity of referential constraints or check constraints is not clear.
- In a system in which the `pd_check_pending` operand was omitted in the system definition, HiRDB was upgraded to version 07-03 or later and the integrity of referential constraints or check constraints is not clear.
- Table manipulation by the user must be regulated temporarily.
- Because a constraint violation was detected during integrity checking, the check pending status was released forcibly, the data resulting in the constraint violation during SQL execution was corrected, and now integrity checking is to be performed again.

Forced release of check pending status

This function forces a table into the non-check pending status (releases the check pending status). You specify `-k release` to use this function.

Forced release of check pending status is used in the following cases:

- The check pending status needs to be forcibly released in order to resume operations, such as when there is no need to perform integrity checking because the user has confirmed the integrity of the constraints.
- A row resulting in a violation of a referential constraint is to be corrected by using SQL statements to update or delete a referencing table's foreign key.
- A row resulting in a violation of a check constraint is to be corrected using SQL statements.

(2) Execution unit of the facility for changing check pending status forcibly

The facility for changing check pending status forcibly can be executed *by table* to update all referential constraints and check constraints defined for a table, or *by constraint* to update only an individual constraint. When the inner replica facility is used to perform processing by table, you can execute the facility for changing check pending status forcibly *by all generations* to process the original RDAREAs and all generations of their replica RDAREAs, *by generation* to process only a single generation, or *by the current RDAREA's generation* to process the generation of the current RDAREA. When the facility is executed by constraint, it is executed by all generations.

When the facility for changing check pending status forcibly is executed, information about the check pending status in data dictionary tables changes. Tables 13-20 and 13-21 show the locations in data dictionary tables where the check pending status changes.

Table 13-20: Locations in data dictionary tables where check pending status changes (when the inner replica facility is not used)

Exe unit	-c option spec	Table constraint		Locations in data dictionary table where check pending status changes			
		Ref const	Check const	SQL_TABLES table		SQL_REFERENTIAL_CONSTRAINS table	SQL_CHEKS table
				CHECK_PEND column	CHECK_PEND2 column	CHECK_PEND column	CHECK_PEND2 column
By table	Not applicable	No	Yes	—	Y	—	Y[1]
		Yes	No	Y	—	Y[1]	—
		Yes	Yes	Y	Y	Y[1]	Y[1]
By constraint	Ref const	Yes	No	Y	—	Y[2]	—
		Yes	Yes	Y	—	Y[2]	—
	Check constraint	No	Yes	—	Y	—	Y[2]
		Yes	Yes	—	Y	—	Y[2]

Legend:

Exe unit: Execution unit

-c option spec: -c option specification

Ref const: Referential constraint

Check const: Check constraint

Y: Check pending status changes.

—: Check pending status remains unchanged (the current status is maintained).

1

The check pending status of all referential constraints or check constraints defined for the table is changed.

2

The check pending status of only the constraint specified in the `-c` option is changed.

Table 13-21: Locations in data dictionary tables where check pending status changes (when the inner replica facility is used)

Execution unit	-c option spec	Table constraint		Locations in data dictionary table where check pending status changes			
		Ref const	Check const	SQL_TABLES table		SQL_REFERENTIAL_CONSTRAINS table	SQL_CHEKS table
				CHECK_PEND column	CHECK_PEND2 column	CHECK_PEND column	CHECK_PEND2 column
By all generations, by generation, or by current RDAREA generation	Not applicable	No	Yes	—	Y	—	Y[1]
		Yes	No	Y	—	Y[1]	—
		Yes	Yes	Y	Y	Y[1]	Y[1]
By constraint	Ref const	Yes	No	Y	—	Y[2]	—
		Yes	Yes	Y	—	Y[2]	—
	Check const	No	Yes	—	Y	—	Y[2]
		Yes	Yes	—	Y	—	Y[2]

Legend:

-c option spec: -c option specification

Ref const: Referential constraint

Check const: Check constraint

Y: Check pending status changes.

— : Check pending status remains unchanged (the current status is maintained).

1

The check pending status of all referential constraints or check constraints defined for the table is changed.

2

The check pending status of only the constraint specified in the `-c` option is changed.

Tables 13-22 and 13-23 show the locations of table information in an RDAREA where check pending status changes.

Table 13-22: Locations of table information in an RDAREA where check pending status changes (when the inner replica facility is not used)

Execution unit	**-c option spec**	**Table constraint**		**Locations of table information in an RDAREA where check pending status changes**	
		Referential constraint	**Check constraint**	**Referential constraint status**	**Check constraint status**
By table	Not applicable	No	Yes	—	Y
		Yes	No	Y	—
		Yes	Yes	Y	Y
By constraint	Referential constraint	Yes	No	Y	—
		Yes	Yes	Y	—
	Check constraint	No	Yes	—	Y
		Yes	Yes	—	Y

Legend:

-c option spec: -c option specification

Y: Check pending status changes. If there is no target replica RDAREA, Y becomes —.

—: Check pending status remains unchanged (the current status is maintained).

Table 13-23: Locations of table information in an RDAREA where check pending status changes (when the inner replica facility is used)

Execution unit	Const with -c option spec	Table constraint		Locations of table information in RDAREA where check pending status changes[2]					
		Ref const	Check constr	Generation 0 (original RDAREA)		Generation 1 (replica RDAREA)		Generation *n* (replica RDAREA)	
				Ref const status	Check const status	Ref const status	Check const status	Ref const status	Check const status
By all generations	N/A	No	Yes	—	Y	—	Y	—	Y
		Yes	No	Y	—	Y	—	Y	—
		Yes	Yes	Y	Y	Y	Y	Y	Y
By generation[1]	N/A	No	Yes	—	—	—	Y	—	—
		Yes	No	—	—	Y	—	—	—
		Yes	Yes	—	—	Y	Y	—	—
By current RDAREA generation	N/A	No	Yes	—	—	—	—	—	Y
		Yes	No	—	—	—	—	Y	—
		Yes	Yes	—	—	—	—	Y	Y
By constraint	Ref const	Yes	No	Y	—	Y	—	Y	—
		Yes	Yes	Y	—	Y	—	Y	—
	Check const	No	Yes	—	Y	—	Y	—	Y
		Yes	Yes	—	Y	—	Y	—	Y

Legend:

Const with -c option spec: Constraint with -c option specified

N/A: Not applicable

Ref const: Referential constraint

Check const: Check constraint

Y: Check pending status changes. If there is no target replica RDAREA, Y becomes —.

—: Check pending status remains unchanged (the current status is maintained).

[1] `-q 1` is displays (generation 1).

[2] This applies to all RDAREAs that store the table (original and replica RDAREAs).

(a) By table

When the facility is executed by table, it changes all the defined referential constraints or check constraints. You specify the `-t` option to execute the facility by table.

You use execution by table in the following case:

- `pdconstck` is to be executed only once to change the check pending status of all referential constraints and check constraints defined for the table.

When the inner replica facility is used, `pdconstck` can be executed by table as well as in the following units:

By all generations

> Among all referential constraints and check constraints defined for the table, `pdconstck` changes the check pending status for all generations of the original and replica RDAREAs that store the table. The criteria are the same as for execution by table. You specify `-q all` to execute `pdconstck` by all generations.

By generation

> Among all referential constraints and check constraints defined for the table, `pdconstck` changes the check pending status of the generation specified in the `-q` option for each constraint. You specify the target generation number in the `-q` option to execute `pdconstck` by generation.
>
> You use execution by generation when you are changing the check pending status of a specific generation.

By the current RDAREA's generation

> Among all referential constraints and check constraints defined for the table, `pdconstck` changes the check pending status of the current RDAREA's generations for each constraint. You omit the `-q` option to execute `pdconstck` by the current RDAREA's generation.
>
> You use execution by the current RDAREA's generation in order to release the check pending status of only the current RDAREA's generation, such as when the table was placed in check pending status as a result of executing a utility such as `pdload` on the current RDAREA's generation.

(b) By constraint

The facility changes only one referential constraint or check constraint defined for the table. You specify the `-c` option to execute the facility by constraint. When the inner replica facility is used, `pdconstck` changes all generations of the original and replica

RDAREAs that store the table.

You use execution by constraint in order to change the check pending status for an individual constraint.

(3) Forced setting of check pending status

This function places a table, constraints, and RDAREAs in check pending status.

(4) Forced release of check pending status

This function releases a table, constraints, and RDAREAs from check pending status. The setting (whether or not the check pending status can be released) depends on the execution unit of `pdconstck` and the check pending status of the table, constraints, and RDAREAs. The following provides the details for each execution unit.

(a) When the inner replica facility is not used

Forced release of check pending status by table

When `pdconstck` is executed by table, it unconditionally places the target table, constraints, and RDAREAs in non-check pending status.

Forced release of check pending status by constraint

When `pdconstck` is executed by constraint, it places the constraint specified in the `-c` option in non-check pending status. Depending on the check pending status of other constraints, the check pending status of the table or RDAREAs may not change. Table 13-24 shows the changes in the check pending status.

Table 13-24: Changes in the check pending status by constraint (forced release of check pending status)

Condition		Check pending status (table and RDAREA)
A constraint that is not specified in the `-c` option is defined	There is a constraint in check pending status	No change
	All constraints are in non-check pending status	Non-check pending status
Only the constraint specified in the `-c` option is defined for the table		Non-check pending status

(b) When the inner replica facility is used

Forced release of check pending status by all generations

When `pdconstck` is executed by all generations, it unconditionally places the target table, constraints, and RDAREAs in non-check pending status.

Forced release of check pending status by generation or the current RDAREA's generation

When `pdconstck` is executed by generation or by the current RDAREA's generation, it places the generation of the RDAREA specified in the `-q` option in non-check pending status. Depending on the check pending status of other generations of the RDAREAs, `pdconstck` places the table and constraints in non-check pending status. Table 13-25 shows the changes in check pending status for the table and constraints.

Table 13-25: Changes in check pending status by generation or by the current RDAREA's generation (forced release of check pending status)

Condition	Change of check pending status	
Check pending status of RDAREA whose generation is not specified in the `-q` option	**Table check pending status**	**Constraint check pending status**
There is a generation in check pending status	No change	No change
All generations are in non-check pending status	Non-check pending status	Non-check pending status

Forced release of check pending status by constraint

When `pdconstck` is executed by constraint (by all generations), the processing is the same as when the inner replica facility is not used.

13.3 Examples

This section presents four examples of `pdconstck`.

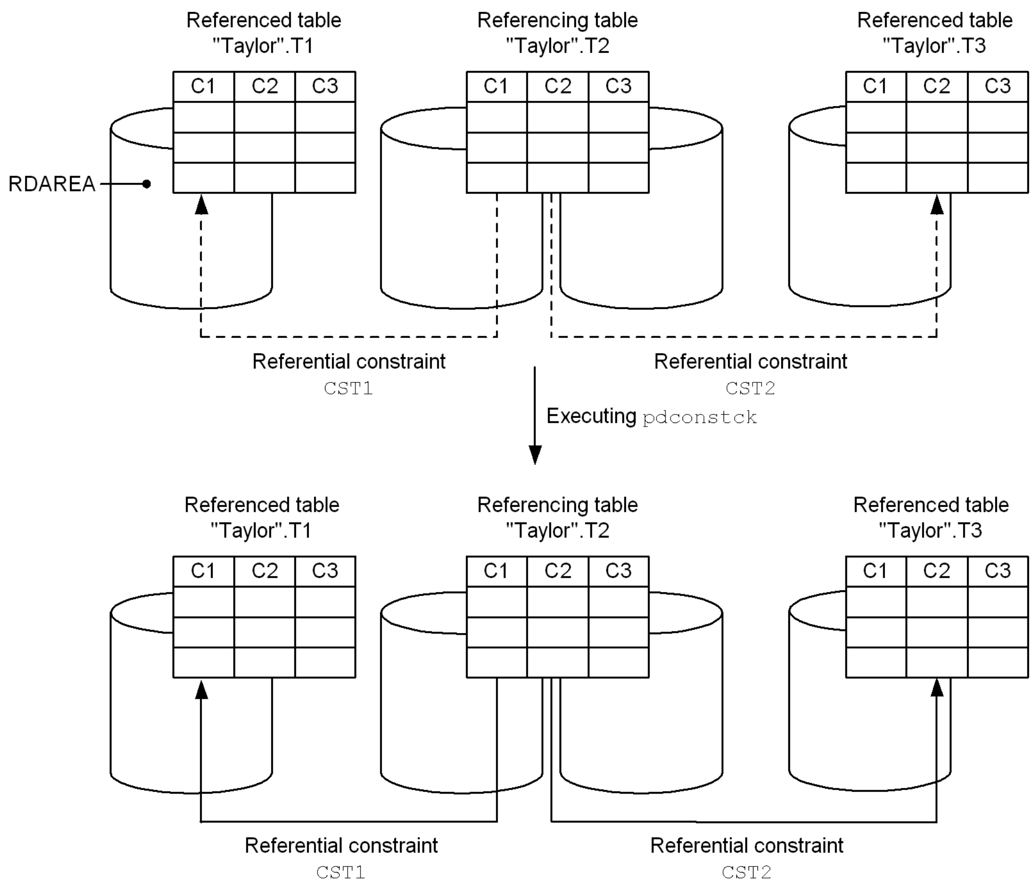
(1) Integrity checking by table

This example performs integrity checking on all constraints (`CST1` and `CST2`) that have been defined for table `T2`. The example assumes that table `T2` is in check pending status.

- Table definitions

```
T1:  CREATE TABLE T1(C1 INT PRIMARY KEY)
T3:  CREATE TABLE T3(C2 INT PRIMARY KEY)
T2:  CREATE TABLE T2(C1 INT, C2 INT,
                    CONSTRAINT CST1 FOREIGN KEY (C1) REFERENCES T1,
                    CONSTRAINT CST2 FOREIGN KEY (C2) REFERENCES T3)
```

Overview



Legend:

- - - -► : Check pending status

——► : Non-check pending status

Command

```
pdconstck -k check -t T2
```

Explanation

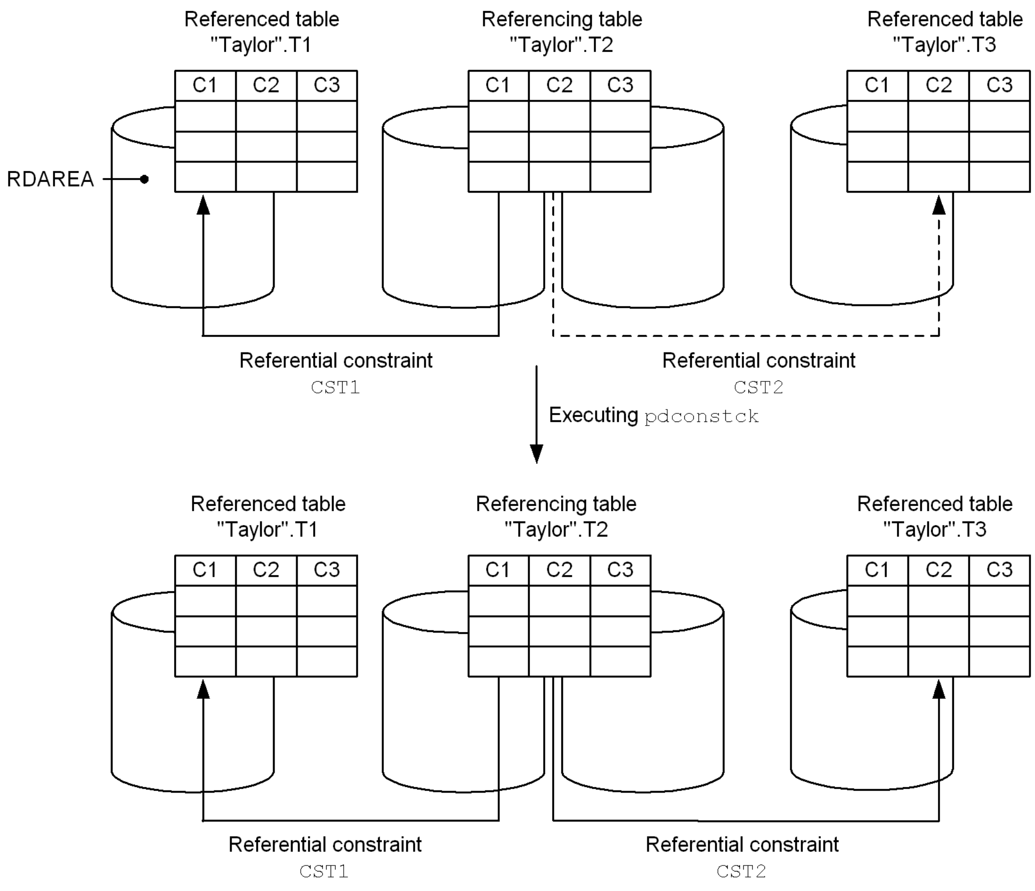
`-k check`: Executes the integrity check facility.

`-t T2`: Specifies the name of the table that is to be the target of integrity checking.

(2) Integrity checking by constraint

This example performs integrity checking on constraint `CST2` defined for table `T2`. The table definitions are the same as in (1), and the example assumes that table `T2` is in check pending status.

Overview


Referenced table "Taylor".T1
Referencing table "Taylor".T2
Referenced table "Taylor".T3
C1 C2 C3
RDAREA
Referential constraint CST1
Referential constraint CST2
Executing pdconstck
Referenced table "Taylor".T1
Referencing table "Taylor".T2
Referenced table "Taylor".T3
Referential constraint CST1
Referential constraint CST2

Legend:

- - - -► : Check pending status

———► : Non-check pending status

Command

```
pdconstck -k check -c CST2
```

Explanation

`-k check`: Executes the integrity check facility.

`-c CST2`: Specifies the name of the constraint that is to be the target of integrity checking.

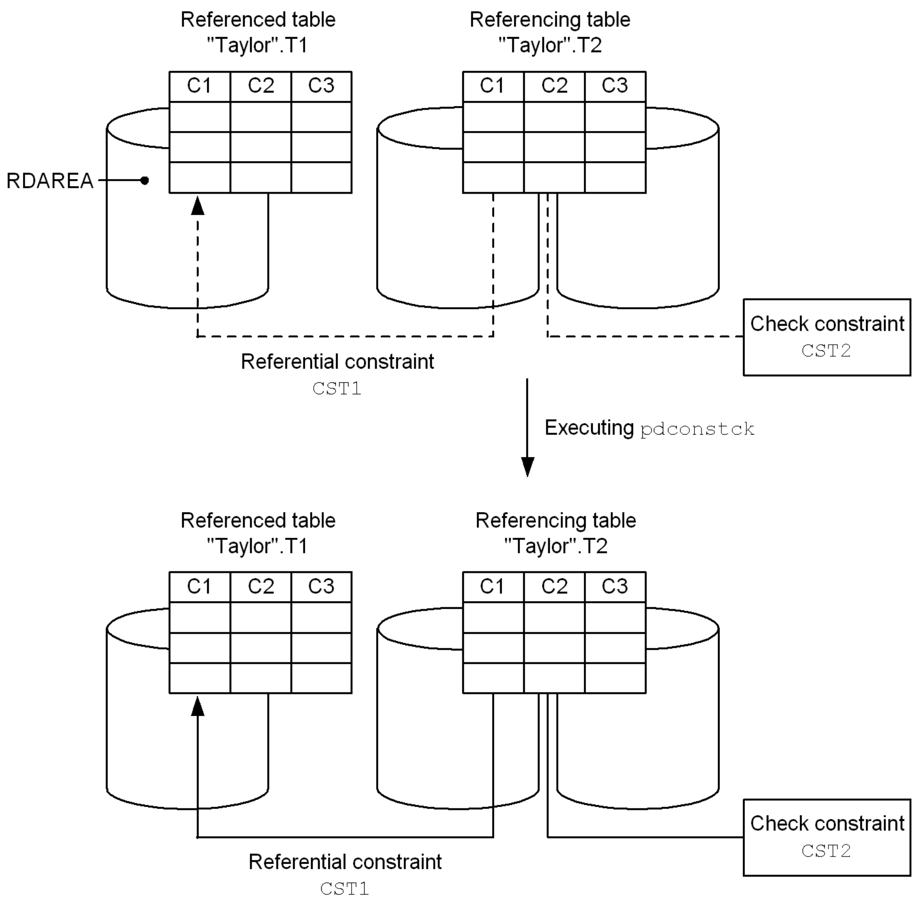
(3) Forced setting of check pending status by table

This example performs integrity checking on all constraints (`CST1` and `CST2`) that have been defined for table `T2`. The example assumes that table `T2` is in non-check pending status.

- Table definitions

```
T1:  CREATE TABLE T1(C1 INT PRIMARY KEY)
T2:  CREATE TABLE T2(C1 INT, C2 INT,
                       CONSTRAINT CST1 FOREIGN KEY (C1) REFERENCES T1,
                       CONSTRAINT CST2 CHECK (c2 > 10))
```

Overview



Legend:

- - - -► and - - - - : Check pending status

——► and —— : Non-check pending status

Command

```
pdconstck -k set -t T2
```

Explanation

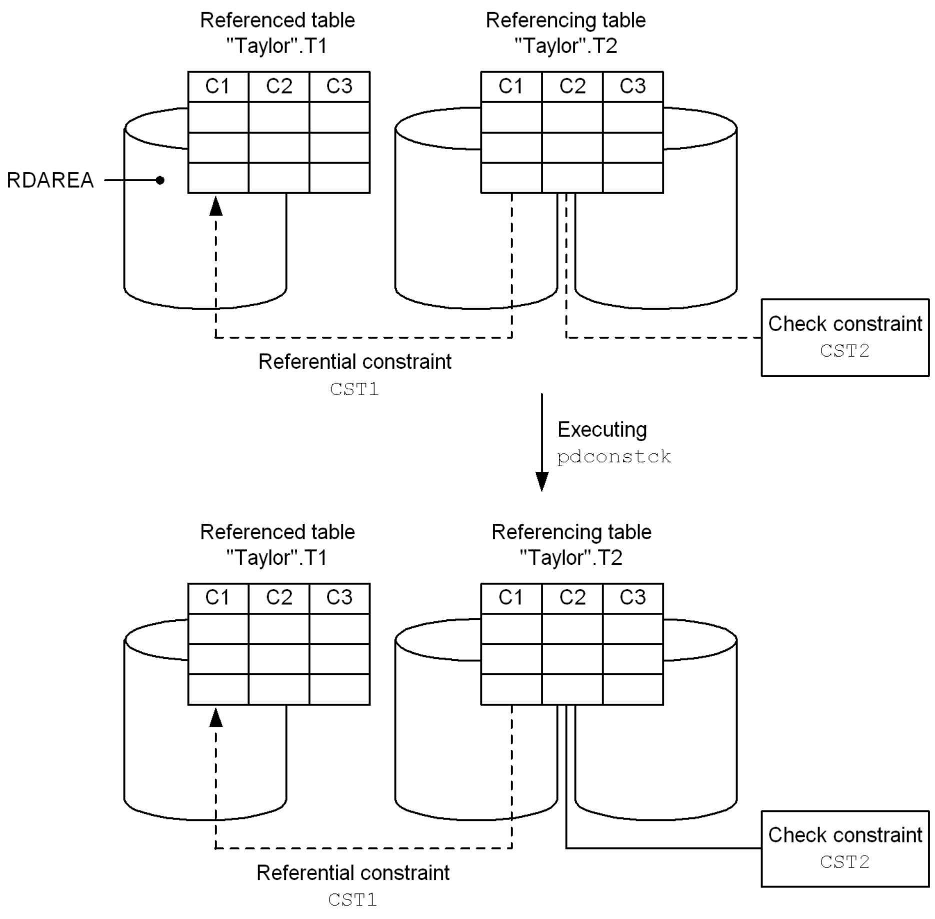
`-k set`: Executes forced setting of check pending status.

`-t T2`: Specifies the name of the table that is to be forcibly placed in check pending status.

(4) Forced release of check pending status by constraint

This example forcibly releases constraint `CST2` for table `T2` from check pending status. The table definitions are the same as in (3), and the example assumes that constraint `CST2` is in check pending status.

Overview



Legend:

----► and ---- : Check pending status

——► and —— : Non-check pending status

Command

```
pdconstck -k release -c CST2
```

Explanation

`-k release`: Executes forced release of check pending status.

`-c CST2`: Specifies the name of the constraint that is to be forcibly released from check pending status.

13.4 Command format

13.4.1 Format

This section explains the format of the `pdconstck` command. In the following table, each number corresponds to the number assigned to each option.

No.	Format
1	`pdconstck -k {check|set|release}`
2	`{-t [`*authorization-identifier*`.]`*table-identifier*
3	`|-c [`*authorization-identifier*`.]`*constraint-name*`}`
4	`[-u` *authorization-identifier*`]`
5	`[-o` *process-results-file-name*`]`
6	`[-q {`*generation-number*`|all}]`
7	`[-w` *maximum-number-of-violation-key-values-that-can-be-output*`]`
8	`[-x` *integrity-checking-monitoring-interval*`]`

13.4.2 Options

(1) -k {check|set|release}

Specifies the function that `pdconstck` is to perform:

check

Executes the integrity check facility.

set

Executes the facility for changing check pending status forcibly (forced setting of check pending status).

release

Executes the facility for changing check pending status forcibly (forced release of check pending status).

(2) -t [authorization-identifier.]table-identifier

～ <identifier>

Specifies the table identifier of the referencing table or check constraint table that is to be processed.

If the authorization identifier is omitted, the authorization identifier of the user who connected to HiRDB is assumed. For details about the authorization identifier of the user who connected to HiRDB, see *13.1(2) Authorization identifier during execution of pdconstck*.

Criteria

Specify this option when you execute the integrity check facility or the facility for changing check pending status forcibly by table.

Rules

1. You can specify only a table for which referential constraints (foreign keys) or check constraints have been defined.
2. A view table cannot be specified.
3. If an authorization identifier or table identifier is enclosed in double quotation marks (`"`), the system treats it as being case sensitive; otherwise, the system treats it as all uppercase letters. If you are using `sh` (Bourne shell), `csh` (C shell), or `ksh` (Korn shell), you must enclose the entire identifier in single quotation marks (`'`).

(3) -c [authorization-identifier.]constraint-name

~ <identifier>

If the authorization identifier is omitted, the authorization identifier of the user who connected to HiRDB is assumed. For details about the authorization identifier of the user who connected to HiRDB, see *13.1(2) Authorization identifier during execution of pdconstck*.

Criteria

Specify this option when you execute the integrity check facility or the facility for changing the check pending status forcibly by constraint.

Rules

1. If an authorization identifier or constraint name is enclosed in double quotation marks (`"`), the system treats it as being case sensitive; otherwise, the system treats it as all uppercase letters. If you are using `sh` (Bourne shell), `csh` (C shell), or `ksh` (Korn shell), you must enclose the entire identifier or name in single quotation marks (`'`).

(4) -u authorization-identifier

~ <identifier>

Specifies the authorization identifier used to connect to HiRDB.

When this option is specified, the system displays a message requesting entry of a password. If no password is required, enter the null in response to the message. For details about the default authorization identifier and password that are assumed when this option is omitted, see *13.1(2) Authorization identifier during execution of pdconstck*.

Criteria

Specify this option to connect to HiRDB using a different authorization identifier from the one specified in the `PDUSER` environment variable.

Rules

1. If an authorization identifier is enclosed in double quotation marks ("), the system treats it as being case sensitive; otherwise, the system treats it as all uppercase letters. If you are using `sh` (Bourne shell), `csh` (C shell), or `ksh` (Korn shell), you must enclose the entire identifier in single quotation marks (').

Notes

You must specify `PDUSER` if you are executing the utility in the background with `&` attached by the shell, or in a remote shell environment in which a password cannot be entered. You must specify `PDUSER` when you are executing this utility in an environment in which a password cannot be entered.

(5) -o process-results-file-name

~ <path name>

Specifies the name of the process results file to which the processing results are to be output.

Criteria

Specify this option to create the process results file in the directory of your choice.

Rules

1. The specified path must be to the host where `pdconstck` is executed.
2. Write privileges must have already been granted by the executor of `pdconstck` for the file specified by the path name (high-order directory).

Default value

When this option is omitted, the process results file is created as follows:

- Directory in which the process results file is created

  `/tmp` directory
- Name of the process results file

  `CONSTCK-REPORT-`*table-name-xxxxxxxxx*

  table-name

Table identifier specified in the `-t` option or the table identifier for which the constraint specified in the `-c` option was defined (1 to 30 characters)

xxxxxxxxx

Unique characters acquired by an OS function (`tmpnam`) (9 characters)

(6) -q {generation-number|all}

~ <unsigned integer> ((1-10))

When the inner replica facility is used, specifies the generation that is to be processed:

`0`: The original RDAREAs are to be processed.

`1` to `10`: The indicated generation of the replica RDAREAs is to be processed.

`all`: The original RDAREA and all generations of replica RDAREAs are to be processed.

Criteria

Specify this option when you use the inner replica facility and wish to execute the integrity check facility or the facility for changing check pending status forcibly by generation.

Rules

1. You can specify this option only when you use the inner replica facility.
2. When the `-c` option is specified, a generation number cannot be specified (only `all` can be specified).
3. You can specify only the generation number of replicas of table storage RDAREAs that store the table specified in the `-t` option, or of replicas of the table for which the constraint specified in the `-c` option has been defined.
4. The following shows the default value that is assumed when the inner replica facility is used and this option is omitted:

-t option	-c option	Default value
Specified	Omitted	Current RDAREA generation
Omitted	Specified	`all`

(7) -w maximum-number-of-violation-key-values-that-can-be-output

~ <unsigned integer> ((1-30000)) <<60>>

The integrity check facility enables you to output to the process results file the key values that violate constraints. This option specifies the maximum number of such violation key values that can be output per constraint (by generation when the inner replica facility is used).

Criteria

Specify this option when you want to reduce file size by limiting output of key values resulting in a constraint violation.

Rules

1. Duplicated key violation values are not output. However, in the case of a check constraint, if the data types of columns related to a check constraint include a `BLOB` or `BINARY` column, duplicates are not eliminated.
2. Once the number of violation key values reaches the specified value (or the default value), no more violation key values are output to the process results file. When this happens, the utility cancels integrity checking on the corresponding constraint (corresponding generation) and performs integrity checking on the next constraint (generation).

(8) -x integrity-checking-monitoring-interval

~ <unsigned integer> ((0-65535)) <<0>>

Specifies the integrity checking monitoring interval in seconds for a single constraint (single generation). When the time spent on integrity checking on a single constraint (single generation) exceeds this operand's value, `pdconstck` terminates with return code `8`.

Criteria

You should specify this option in the event of a failure such as the one listed below, and it is advisable to monitor execution time in order to check whether or not `pdconstck` has stopped responding:

- Communication failure (communication between `pdconstck` and HiRDB server or communication between HiRDB servers)
- Non-responsive process due to a failure (such as a disk failure)

Rules

The `pdconstck` processing depends on the combination of the `-x` option and the `pd_lck_wait_timeout` operand in the system definition. The following table describes the possible combinations and the `pdconstck` processing.

pd_lck_wait_timeout operand value	-x option value	pdconstck processing
`0`	`0`	Waits until it receives a response from the HiRDB server.
	Other	When the `-x` option value is exceeded, `pdconstck` terminates with return code `8`.

pd_lck_wait_timeout operand value	-x option value	pdconstck processing
Other	0	When the pd_lck_wait_timeout operand value is exceeded, pdconstck terminates with return code 8.
	Less than the pd_lck_wait_timeout operand value	When the -x option value is exceeded, pdconstck terminates with return code 8.
	Same as the pd_lck_wait_timeout operand value	When the pd_lck_wait_timeout operand value is exceeded, pdconstck terminates with return code 8.
	Greater than the pd_lck_wait_timeout operand value	

13.5 pdconstck processing results

This section describes the processing results of `pdconstck`.

(1) pdconstck processing results

The `pdconstck` processing results include the processing results of integrity checking and information about the check pending status. The following is an example of the `pdconstck` processing results:

```
          [1]                                    [2]        [3]
pdconstck VV-RR *** DB VALIDATION CHECK *** yyyy-mm-dd hh:mm:ss
processing-results-of-integrity-checking [4]
information-about-check-pending-status [5]
pdconstck TERMINATED, RETURN CODE=0[6]
```

Explanation

1. HiRDB version number
2. `pdconstck` execution start date
3. `pdconstck` execution start time
4. Processing results of integrity checking
5. Information about check pending status
6. `pdconstck`'s return code

(2) Processing results of integrity checking

The execution results of the integrity check facility are output by `pdconstck` as the integrity checking processing results. This information is output when `-k check` is specified.

The following shows an output example of the processing results of integrity checking:

```
** DB VALIDATION CHECK INFORMATION **
processing-results-of-integrity-checking-on-referential-constraints [1]
processing-results-of-integrity-checking-on-check-constraints [2]
```

Explanation

1. Processing results of integrity checking on referential constraints
2. Processing results of integrity checking on check constraints

   These are the processing results of executing integrity checking on check constraints. This information is output when the `-t` option is specified and check constraints have been defined for the table, or when the name of a check constraint is specified in the `-c` option.

(a) Processing results of integrity checking on referential constraints

The processing results of integrity checking on referential constraints are output in alphanumeric order of the constraint names when the `-t` option is specified and referential constraints have been defined for the table, or when the name of a referential constraint is specified in the `-c` option.

The following shows an output example of the processing results of integrity checking on a referential constraint:

```
      ** REFERENTIAL CONSTRAINT INFORMATION **
      CONSTRAINT    = SCHEMA1.CONST1 [1]
      REFERENCED  TABLE = SCHEMA1.P1 [2]
      REFERENCING TABLE = SCHEMA1.F1 [3]
      [5]              [6]
[4] - NO.  FOREIGN KEY COLUMN NAME
   | ---- ----------------------------
   |  1   COL1
    - 2   COL2

      execution-results  [7]
```

Explanation

1. Name of a constraint (*authorization-identifier*.*constraint-name*)
2. Name of the referenced table (*authorization-identifier*.*table-identifier*)
3. Name of the referencing table (*authorization-identifier*.*table-identifier*)
4. Column information for foreign keys
5. Definition order
6. Column names (in the order that the foreign key component columns are defined)
7. Execution results

   These are the results of integrity checking on the constraint. This information is not output for a constraint whose integrity was not checked.

   When the inner replica facility is not used, only one set of the information items is output; when the inner replica facility is used, as many sets of the information items are output as there are generations on which integrity checking was executed, in ascending order of the original RDAREA generations (from 1 to 10). The following shows an output example of the results of integrity checking on referential constraints, both when there are no violation key values and when there are violation key values.

Execution results when there are no violation key values

```
GENERATION    = 0 [1]
NO ERROR FOREIGN KEY [2]
```

Explanation

1. Number of the generation on which integrity checking was performed

   When the inner replica facility is not used, this information is not output. When `pdconstck` was executed by current RDAREA, `CUR` is output.

2. Indicator that there were no violations foreign key values.

Execution results when there are violation key values

```
     GENERATION    = 0 [1]
     [3]          [4]
[2] - NO.   ERROR FOREIGN KEY
    |----- ----------------
    |   1   1234567890
    |       Taylor
    |   2   777
     -      Young
     PROCESSING DISCONTINUED [5]
     EXIST ERROR FOREIGN KEY [6]
```

Explanation

1. Number of the generation on which integrity was checked

   When the inner replica facility is not used, this information is not output. When `pdconstck` was executed by current RDAREA, `CUR` is output.

2. Information about the violation key values (in the order of the primary key values referenced by the foreign key).

3. Serial numbers of the violation keys (in ascending order starting with 1)

   The violation key values are output in the order of the column names of the foreign key. If there are multiple columns composing the foreign key, only the first key value is output.

4. Violation key values

   A violation key value is displayed in character format.

   In the case of a character data type, only the first 70 bytes are output as a violation key value. A national character that ends in byte 71 is not output.

5. Indicator that integrity checking was cancelled.

6. Indicator that violation foreign key values were found.

(b) Processing results of integrity checking on check constraints

The processing results of integrity checking on check constraints is output when the `-t` option is specified and check constraints have been defined for the table, or when the name of a check constraint is specified in the `-c` option.

The following shows an output example of the processing results of integrity checking on a check constraint:

```
       ** CHECK CONSTRAINT INFORMATION **
       CONSTRAINT = SCHEMA1.CONST2 [1]
       TABLE      = SCHEMA1.TABLE1 [2]
       [4]            [5]
[3] -  NO.    CHECK KEY COLUMN NAME
    | ---- ------------------------------
    |  1   COL1
     - 2   COL2

       execution-results [6]
```

Explanation

1. Name of a constraint (*authorization-identifier*.*constraint-name*)
2. Name of the table (*authorization-identifier*.*table-identifier*)
3. Column information in the search conditions (in the order stored in the `SQL_CHECK_COLUMNS` data dictionary table)
4. Serial numbers assigned to the column information
5. Column names
6. Execution results

   These are the results of integrity checking on the constraint. This information is not output for constraints whose integrity was not checked.

   When the inner replica facility is not used, only one set of information items is output; when the inner replica facility is used, as many sets of information items are output as there are generations on which integrity checking was executed, in ascending order of the original RDAREA generations (from 1 to 10).

The following shows an output example of the results of integrity checking on check constraints, both when there are no violation key values and when there are violation key values.

Execution results when there are no violation key values

```
GENERATION    = 0 [1]
NO ERROR CHECK KEY [2]
```

Explanation

1. Number of the generation on which integrity was checked

   When the inner replica facility is not used, this information is not output. When `pdconstck` was executed by current RDAREA, `CUR` is output.

2. Indicator that there were no violation foreign key values.

Execution results when there are violation key values

```
     GENERATION    = 0 [1]
     [3]          [4]
[2] - NO.   ERROR CHECK KEY
    |----- ----------------
    |   1   1234567890
    |       Taylor
    |   2   777
     -      Young
     PROCESSING DISCONTINUED [5]
     EXIST ERROR CHECK KEY [6]
```

Explanation

1. Number of the generation on which integrity was checked

   When the inner replica facility is not used, this information is not output. When `pdconstck` was executed by current RDAREA, `CUR` is output.

2. Information about the violation key values (in the order of integrity checking on check constraints)

3. Serial numbers of the violation keys (in ascending order starting with 1)

4. Violation key values

   If a key value is the null value, `*NULL*` is output.

   If the data type of the column that constitutes the check constraint is `BLOB` or `BINARY`, only the first 35 bytes of the violation key value are displayed (in hexadecimal). If the data type of the column that constitutes the check constraint is neither `BLOB` nor `BINARY`, the violation key value is displayed in the data format.

   In the case of a character data type, only the first 70 bytes are output as a violation key value. A national character that ends in byte 71 is not output.

5. Indicator that integrity checking was cancelled.
6. Indicator that violation foreign key values were found.

(3) Information about check pending status

Regardless of the `-k` option's value, the check pending statuses of the table and check constraints are output. An output example of check pending status information is shown below. Note that the check pending status of each RDAREA is not output. For the check pending status of each RDAREA, check `pddbst`'s status analysis by RDAREA (logical analysis) or status analysis by table.

```
 ** CHECK PENDING STATUS INFORMATION **
                                            [2]           [3]
 TABLE                              CONSTRAINT TYPE PENDING STATUS
 ---------------------------------------- ---------------
---------------
 SCHEMA1.T5 [1]
                                          REFERENCE        PENDING
                                           CHECK           RELEASE
                                           [2]           [3]
 CONSTRAINT                         CONSTRAINT TYPE PENDING STATUS
 ---------------------------------------- ---------------
---------------
 SCHEMA1.CONST1 [4]                       REFERENCE        PENDING
 SCHEMA2.CONST2 [4]                        CHECK           RELEASE
```

Explanation

1. Table name
2. Type of constraint:

   `REFERENCE`: Referential constraint (if no referential constraints have been defined, this information is not output)

`CHECK`: Check constraint (if no check constraints have been defined, this information is not output)

3. Check pending status:

   `PENDING`: Check pending status

   `RELEASE`: Non-check pending status

4. Constraint name:

   Outputs the names of the referential constraints and then the check constraints that have been defined for this table, in ascending alphanumeric order of the constraint names.

13.6 Notes

1. Table 13-26 lists and describes the return codes of `pdconstck` and explains the actions to be taken.

   Table 13-26: Return codes of pdconstck and actions to be taken

Return code	Description	Action
`0`	Terminated normally.	None
`4`	Terminated normally, but there was a constraint that lacked integrity as a result of integrity checking (this includes cases such as when integrity checking on constraints was cancelled because the number of violation key values exceeded the specified maximum).	Correct the integrity of the constraints resulting in check pending status and then re-execute.
`8`	Terminated abnormally.	If necessary, eliminate the cause of the error and then re-execute.

   Note

   If an error occurs during `pdconstck`'s termination processing after return code `0` or `4` has been output with the `KFPL50001-I` message, the return code remains `0` or `4`, not `8`.

2. When `pdconstck` is executed while the security audit facility is being used, the audit trails that are acquired depend on the return code:

   When the return code is `0` or `4`:

   Audit trails are acquired if `SUCCESSFUL` or `ANY` is specified for `WHENEVER` in `CREATE AUDIT`.

   When the return code is `8`:

   Audit trails are acquired if `UNSUCCESSFUL` or `ANY` is specified for `WHENEVER` in `CREATE AUDIT`.

3. If you selected `utf-8` as the character encoding in the `pdsetup` command, a BOM is not added to the file that is output by `pdconstck`.

Chapter

14. Statistics Analysis Utility (pdstedit)

This chapter explains the statistics analysis utility (`pdstedit`) that edits statistical information, such as information about HiRDB system activities.

This chapter contains the following sections:

14.1 Overview

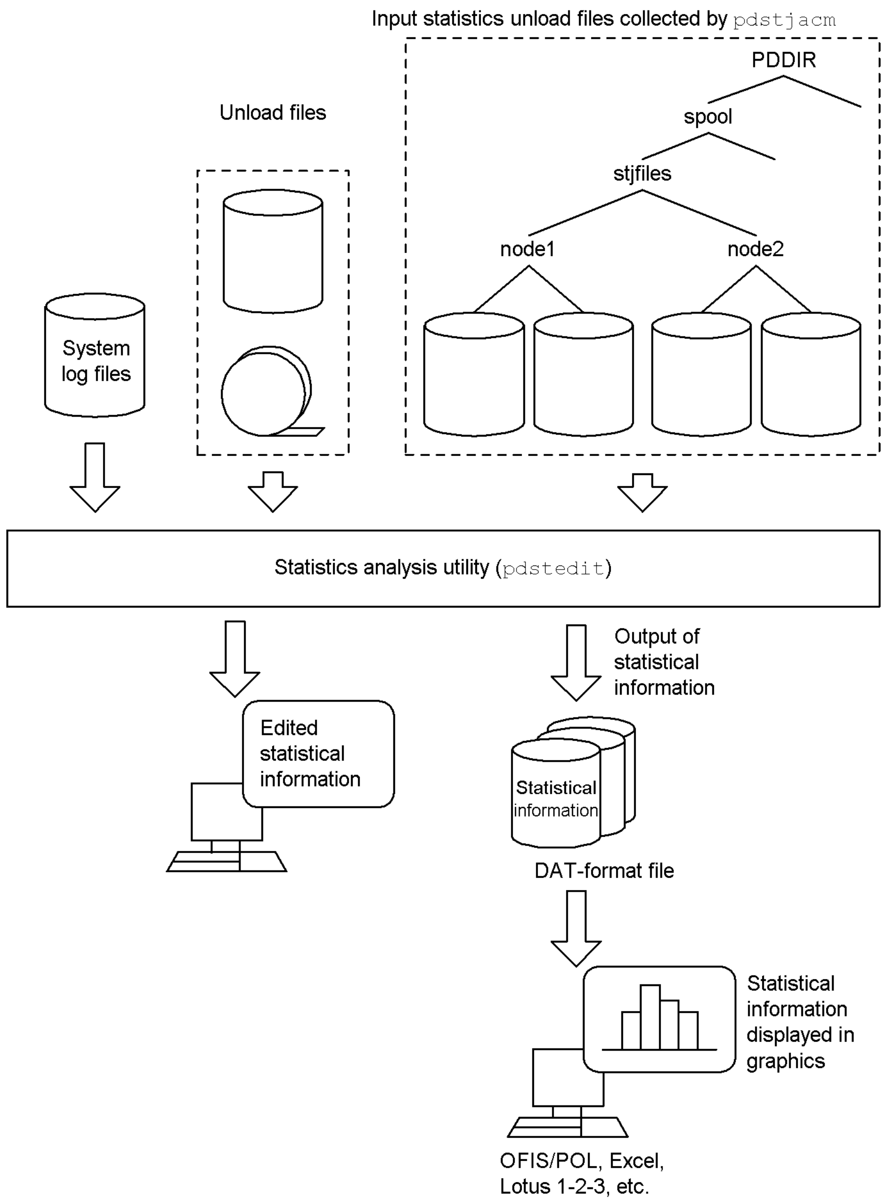
Executor: HiRDB administrator

The statistics analysis utility reads information input from statistics unload files and system log files and edits statistical information.

The statistics analysis utility enables needed statistical information to be edited and output by specifying editing beginning and ending points. The user can obtain the activity status of the HiRDB system by analyzing this statistical information. The utility also outputs the statistical information editing results to the standard output, and it outputs the unedited statistical information to a DAT-format file. The user can create desired statistical reports from the DAT-format file.

Figure 14-1 provides an overview of the statistics analysis utility (`pdstedit`).

Figure 14-1: Overview of the statistics analysis utility (pdstedit)


Input statistics unload files collected by pdstjacm
PDDIR
spool
stjfiles
node1
node2
Unload files
System log files
Statistics analysis utility (pdstedit)
Output of statistical information
Edited statistical information
Statistical information
DAT-format file
Statistical information displayed in graphics
OFIS/POL, Excel, Lotus 1-2-3, etc.

Note: `pdstjacm` can collect only unload statistics log files as the statistics input unload files.

(1) Base files for statistical information

The statistics analysis utility uses the following three files as its input files:

- Unload statistics log file

  This is an unloaded statistics log file. The OS command `cp` or the HiRDB-provided shell script (`pdstjacm`) is used to create an unload statistics log file.

- Unload log file

  This is a system log file unloaded by the `pdlogunld` command.

- System log file group

  This is the logical unit of the system log files. The input is the file groups containing the physical files that can be referenced from the host that starts the statistics analysis utility.

(2) Types of statistical information

Table 14-1 describes the types of statistical information that the statistics analysis utility can edit. You collect statistical information (except `CONNECT` and `DISCONNECT` information) by specifying the `pdstbegin` operand in the system definition or by executing the `pdstbegin` command. To collect statistical `CONNECT` and `DISCONNECT` information, you specify the `pdhibegin -k cnc` operand in the system definition.

Table 14-1: Overview of statistical information output by statistics analysis utility

Type of statistical information	Overview
System activity statistical information	Edits and outputs activity information for processes, RPC, and logs in HiRDB systems by HiRDB system or by server.
UAP statistical information[1]	Edits by editing interval information on UAPs, such as the number of selection rows and the number of times each SQL statement is executed, and outputs the information by UAP or service.
SQL statistical information[1]	Edits and outputs by UAP or service information about SQL statements issued, such as the number of selection rows, the number of times a work table was created, and the number of back-end servers that issued an SQL split command.
Global buffer pool statistical information[2]	Edits by editing interval information about global buffer accesses, such as the buffer hits rate and the number of real I/O operations, and outputs the information by server or global buffer.
Statistical information on HiRDB files for database manipulation[2]	Edits by editing interval information on HiRDB file accesses, such as the number of synchronous I/O operations and the number of I/O errors, and outputs the information by server, HiRDB file, or RDAREA.

Type of statistical information	Overview
Deferred write processing statistical information	Edits by editing interval information on deferred write processing, such as the number of deferred write operations, the cause of each operation, and the concurrency level of I/O operations, and outputs the information by server.
Index statistical information[3]	Edits by editing interval information on indexes, such as index key lock information and index split information, for the statistics log and the system log, and outputs the information by server or index. This information cannot be output to a DAT-format file.
SQL static optimization information	Outputs SQL static optimization information. This information can be output to a DAT-format file only.
SQL dynamic optimization information	Outputs SQL dynamic optimization information. This information can be output to a DAT-format file only.
SQL object execution information	Outputs SQL object execution information. This information can be output to a DAT-format file only
SQL statement statistical information	Outputs issued data manipulation SQL, definition SQL, and `LOCK` statements and SQL information. This information can be output to a DAT-format file only.
`CONNECT`/`DISCONNECT` statistical information[4]	Outputs `CONNECT` and `DISCONNECT` information. This information can be output to a DAT-format file only.
SQL object transfer statistical information	Outputs SQL object transfer information. This information can be output to a DAT-format file only.
Foreign server operation statistical information	Outputs foreign server activity information. This information can be output to a DAT-format file only.
Foreign server utilization statistical information	Outputs foreign server utilization status. This information can be output to a DAT-format file only.

[1] UAP and SQL statistical information may not agree, due to differences in the timing of entry of the `pdstbegin` and `pdstend` commands. Also, the number of SQL statements actually issued by the UAP and the number displayed may not agree.

[2] When a synchronization point is not generated within the time specified by the `-t` option or when only one synchronization point is generated, information for that interval of time is not output. A longer time interval should be specified in the `-t` option in this case.

[3] Index split information, which is one of the index statistical information items, is collected from the unload log files or system log files. To edit index split information, the file groups of the unload log files or the system log files must be used as the input. Other information is obtained from unload statistics log files.

4 `CONNECT` and `DISCONNECT` statistical information is collected from unload log files or system log file groups. Therefore, use unload log files or system log file groups as the input to the statistics analysis utility. If the `pdhibegin -k cnc` operand is specified in the system definition, this statistical information is output to the single-server or to a server with a front-end server.

(3) Execution conditions of the statistics analysis utility

1. You can execute the statistics analysis utility whether or not HiRDB is active.
2. You can execute the statistics analysis utility on any server machine.

14.2 Command format

14.2.1 Format

This section describes the format of the `pdstedit` command. In the following table, each number corresponds to the number assigned to each option.

No.	Format
1	`pdstedit [-k` *edit-item*`[,`*edit-item*`]...]`
2	`[-m` *interval*`]`
3	`[-t [`*start-time*`][,`*end-time*`]]`
4	`[-u` *UAP-name*`[,`*UAP-name*`]...]`
5	`[-x` *host-name*`[,`*host-name*`]...]`
6	`[-s` *server-name*`[,`*server-name*`]...]`
7	`[-f` *foreign-server-name*`[,`*foreign-server-name*`]...]`
8	`[-i {` *input-statistics-unload-file-name* `|`*HiRDB-file-system-area-name* `|`*input-statistics-unload-file-storage-directory-name*`}]`
9	`[-o` *DAT-format-file-output-destination-directory- name*`]`
10	`[-w` *work-file-directory-name*`]`
11	`[-d` *control-statement-file-name*`]`
12	`[-b]`
13	`[-e` *extended-format-specification-value*`[,`*extended-format-specification-value*`]]`

14.2.2 Options

(1) -k edit-item

~ <<`sys`>>

Specifies the information that is to be edited.

If you specify `sop`, `dop`, `pcd`, `obj`, `sqh`, `cnc`, `fsv`, or `hba`, make sure that the `-o` option is also specified (otherwise, an error results).

When `sop`, `dop`, `pcd`, `obj`, `sqh`, `cnc`, `fsv`, or `hba` is specified, the utility outputs statistical information to a DAT-format file, but it does not edit the statistical information. If you specify `all` and omit the `-o` option, the utility continues processing without detecting an error.

`sys`: System activity statistical information

`svr`: System activity statistical information by server

`uap`: UAP statistical information

`sql`: SQL statistical information

`buf`: Global buffer pool statistical information

`fil`: Statistical information on HiRDB files for database manipulation

`dfw`: Deferred write processing statistical information

`idx`: Index statistical information

`sop`: SQL static optimization information

`dop`: SQL dynamic optimization information

`pcd`: SQL object execution information

`obj`: SQL object transfer statistical information

`sqh`: SQL statement statistical information

`cnc`: `CONNECT`/`DISCONNECT` statistical information

`fsv`: Foreign server operation statistical information

`hba`: Foreign server utilization statistical information

`all`: All the above (if `sop`, `dop`, `pcd`, `obj`, `sqh`, `cnc`, `fsv`, or `hba` is specified, the statistical information is output to a DAT-format file only)

(2) -m interval

~ <unsigned integer> ((1-1440)) <<60>>

Specifies in minutes the interval at which totaled values are to be output for the selected types of statistical information. Totaled values for each selected type of statistical information are output at this interval.

(3) -t [start-time][,end-time]

Specifies a log record output start time and end time as the period for which statistical information is to be output. The permitted value range for both the start time and the end time is from 00:00:00 on January 1, 1970, to the current time on the current date.

When this option is omitted, all statistical information in the input statistics unload files and system log files is edited.

When the start time is omitted, editing is from the beginning of the input statistics unload files or system log files to the specified end time. When the end time is omitted, editing is from the specified start time to the end of the statistics input unload files or system log files.

When the option flag is specified, a start time or an end time (or both) must be specified. A time is specified in the format *hhmmss[MMDD[YYYY]]*:

hh: Hour ((00-23))

mm: Minute ((00-59))

ss: Second ((00-59))

MM: Month ((01-12)) (if omitted, the current month is assumed)

DD: Date ((01-31))

> If omitted, the current date is assumed. If an invalid date specified (such as specifying 31 for the month that has only 30 days), the utility carries over the extra days to the next month.
>
> For example, the utility interprets a specification of `00000006312000` as 00:00:00 on July 1, 2000.

YYYY: Year ((1970-9999)) (if omitted, the current year is assumed)

(4) -u UAP-name[,UAP-name]...

~ <character string> ((1-30))

Specifies the names of UAPs that are to be subject to information editing (ID name of a UAP as specified in `PDCLTAPNAME` in the client environment definition) when the editing is to be of UAP or SQL statistical information, foreign server operation statistical information, or foreign server utilization statistical information.

A maximum of 16 UAP names can be specified. When this option is omitted, the utility edits UAP or SQL statistical information according to the range specified with the `-t` option.

(5) -x host-name[,host-name]...

~ <identifier> ((1-32))

Specifies the names of the hosts that are to be subject to information editing. A maximum of 32 host names can be specified. When this option is omitted, all hosts become subject to information editing.

(6) -s server-name[,server-name]...

~ <identifier> ((1-8))

Specifies the names of the servers that are to be subject to information editing. A maximum of 32 server names can be specified. When this option is omitted, all servers become subject to information editing.

(7) -f foreign-server-name[,foreign-server-name]...

~ <identifier> ((1-30))

Specifies the names of foreign servers in order to edit statistical information about their operation (foreign server operation statistical information) or utilization (foreign server utilization statistical information).

You can specify a maximum of 32 foreign server names. When this option is omitted, all foreign servers are subject to editing of statistical information.

(8) -i {input-statistics-unload-file-name|HiRDB-file-system-area-name|input-statistics-unload-file-storage-directory-name}

Specifies the input statistics unload files that are to be used as the input information for statistical analysis.

The host executing `pdstedit` must be able to reference the specified input statistics unload files. An input statistics unload file means an unload statistics log file or an unload log file.

When this option is omitted, the utility assumes the standard input.

input-statistics-unload-file-name ~ <path name>

Specifies the name of an input statistics unload file.

HiRDB-file-system-area-name ~ <path name> ((up to 165 characters))

Specifies the name of a HiRDB file system area when the desired input statistics unload files are located in a HiRDB file system area.

The utility analyzes all input statistics unload files contained in the specified HiRDB file system area.

input-statistics-unload-file-storage-directory-name ~ <path name>

Specifies the name of a directory containing input statistics unload files.

When a directory name is specified, the files under its subdirectories are subject to editing (if there is a HiRDB file system area, the files contained in the HiRDB file system area are subject to editing).

(a) Notes

1. Specification of relative path

   If you specify a relative path (beginning with a character other than the forward slash (`/`)) in the `-i` option, the utility links the executor's `PDDIR` environment variable to the specified relative path to locate the files or directory subject to analysis. If no such file or directory is found, the utility uses the file under the current directory or the current directory itself.

2. Output mode for unload statistics log files

   You cannot use the statistics analysis utility on a 32-bit-mode HiRDB to edit an unload statistics log file output by a 64-bit-mode HiRDB.

3. File type

   Analysis processing may be placed on hold or skipped depending on the type of file or directory specified in the `-i` option. The following table describes whether or not analysis processing occurs depending on the file type:

Entity specified by -i option	File type		Analysis processing
File	Regular file		Processing continues.
	RAW file		Processing continues.
	Pipe file (FIFO)*		Analysis processing is placed on hold until the specified file is opened in write mode by another process.
	Other		Analysis processing on the specified file is skipped.
Directory	File under the directory	Regular file	Processing continues.
		RAW file	Processing continues.
		Pipe file (FIFO)	Warning message is output.
		Other	Analysis processing on the specified file is skipped.

* You can specify a pipe file (FIFO) when there is only one statistics log file.

(b) Notes on input of unload log files

When you use the `pdlogunld` command to unload a system log file, specify the `-n` option (to prevent the system log status from being changed). If the system log status changes, the system log file may not be unloaded for the following reasons:

- If the automatic log unloading facility is used for a system log file, the system log file is treated as having been already unloaded, and it is not unloaded automatically.
- If a system log file is placed in unload completed status, HiRDB assumes that the system log file can be overwritten, so the file may be overwritten before it is unloaded.

(c) Notes on input of unload log files in HiRDB file system areas in character special files

- When transferring unload log files to a host where `pdstedit` is not executed

It is difficult to collect unload log files in HiRDB file system areas in character special files at a single location because they cannot be transferred to another host by a method such as `rcp`. In such a case, use the following method to transfer unload log files to another host:

1. Use the `pdfbkup` command to create a backup of the HiRDB files in a HiRDB file system area that is subject to analysis.
2. Use a command such as `rcp` to transfer the backup file to the other host where `pdstedit` is to be executed.
3. Use the `pdfmkfs` command to create a HiRDB file system area in a regular file for use as input for statistics analysis. To analyze other files at the same time, create a HiRDB file system area in the same directory as for those files.
4. Use the `pdfrstr` command to restore the contents of the backup file transferred in step 2 into the HiRDB file system area created in step 3.
5. When executing `pdstedit`, specify in the `-i` option the name of the HiRDB file system area restored in step 4 or the name of the directory containing the HiRDB file system area.

- When reading unload log files by the host where `pdstedit` is executed

When multiple files are to be analyzed by `pdstedit`, all the files subject to analysis are placed in a single directory and that directory is specified. However, if an unload log file is located in a HiRDB file system area in a character special file, it is difficult to input it together with other files because it cannot be copied by a command such as `cp`.

In such a case, by symbolically linking the directory containing the files to be analyzed to the HiRDB file system area in the character special file that contains the unload log files to be analyzed, you can analyze all the files without having to copy the latter.

(9) -o DAT-format-file-output-destination-directory-name

~ <pathname> ((1-128))

When a DAT-format file is to be created by collecting statistical information from an input statistics unload file, specifies the name of the directory under which the DAT-format file is to be created.

(10) -w work-file-directory-name

~ <pathname> ((1-128))

Specifies the name of the directory under which work files are to be created for the statistics analysis utility. When this option is omitted, `/tmp` or `/usr/tmp` is assumed. An error results if work files cannot be created in `/tmp` or `/usr/tmp` for some reason.

The statistics analysis utility calls a sort program internally. In this case as well, the sort program creates temporary work files in `/tmp` or `/usr/tmp`. You can use the `TMPDIR` environment variable to change the directory where the temporary work files are created.

If the statistics analysis utility is cancelled during execution, such as by forced or abnormal termination of the process, the temporary work files may remain in the directory. If such temporary work files are not needed, the user should delete them. The following files are created in the directory for temporary work files:

-k option value	Names of files that are created
`sys`	*PID*`.syi`, *PID*`.syo`
`uap`	*PID*`.uai`, *PID*`.uao`
`sql`	*PID*`.sqi`, *PID*`.sqo`
`buf`	*PID*`.bui`, *PID*`.buo`
`fil`	*PID*`.fii`, *PID*`.fio`
`dfw`	*PID*`.dfi`, *PID*`.dfo`
`idx`	*PID*`.ixi`, *PID*`.ixo`, *PID*`.isi`, *PID*`.iso`
`sop`	*PID*`.soi`, *PID*`.soo`
`dop`	*PID*`.doi`, *PID*`.doo`
`pcd`	*PID*`.pci`, *PID*`.pco`
`obj`	*PID*`.obi`, *PID*`.obo`
`sqh`	*PID*`.qhi`, *PID*`.qho`
`cnc`	*PID*`.cni`, *PID*`.cno`
`fsv`	*PID*`.fsi`, *PID*`.fso`
`hba`	*PID*`.hbi`, *PID*`.hbo`

Legend:

PID: Process ID

Note

If `-k all` is specified, all the files listed above are created.

(11) -d control-statement-file-name

~ <pathname>

Specifies the name of the control statements file in which are specified the names of the file groups that store the system log information that is to be input to the statistics analysis utility.

When the `-i` option is not specified, the status changes to the standard input standby status even if the `-d` option is specified.

Following is the format of the control statements file (`file_group` statements); a maximum of 32 control statements can be specified in this file:

```
file_group server-name:file-group-name[,file-group-name]...
```

server-name ~ <identifier> ((1-8))

Specifies the name of the server corresponding to the file groups. When the `-s` option is specified and the server name specified here is different from the server name specified in the `-s` option, the server cannot be analyzed and is ignored. An error will result if the same server name is specified more than once in the control statements file.

file-group-name ~ <identifier> ((1-8))

Specifies a file group of physical files that can be referenced from the host that starts the statistics analysis utility.

A maximum of 200 file group names can be specified.

An error results if a file group at another host is specified.

If any file in a file group on a shared disk is specified, however, that file is subject to analysis.

Rules

1. Specify one control statement per line.
2. Do not specify spaces or tabs between specification values.
3. Any text enclosed between `/*` and `*/` is treated as a comment.
4. If the current file group or any inconsistent file group is specified, the utility outputs a warning message and edits statistical information up to immediately before the error was detected (in the case of a duplicate specification, two warning messages may be output).

   The possible causes of warning errors are as follows:

- End-of-file was detected.
- File contents are invalid.

5. If the specified file group has been initialized but not used, the utility outputs a warning message without editing the statistical information (in the case of a duplicate specification, two warning messages may be output).
6. If a specified file is not a system log file, the utility outputs a warning message without editing the corresponding file group.
7. If the `-i` and `-d` options are both specified, the file group in the control statement file specified in the `-d` option has already been unloaded, and that unload log file is specified in the `-i` option, the utility edits the information in the specified file group twice.

(12) -b

Specifies that a title bar is to be output to the DAT-format file. Specification of this option is ignored if the `-o` option is not specified.

(13) -e extended-format-specification-value[,extended-format-specification-value]

Specifies that the output format is to be changed when a DAT-format file is output. When this option is specified, the `-o` option must also be specified.

You can specify `sec` or `er1` as an extended format specification value.

`sec`:

Specifies that the statistics log acquisition time is to be displayed down to the level of the second for the following statistical information:

- Global buffer pool statistical information
- Statistical information on HiRDB files for database manipulation
- Statistical information on deferred write processing

When this option is omitted, the statistics log acquisition time is output in the format *MM/DD/hh:mm*. When this option is specified, the statistics log acquisition time is output in the format *MM/DD/hh:mm:ss*.

`er1`:

Specifies that only significant digits are to be output for overflow data[1] or uneditable data.[2] When `er1` is specified, one asterisk (`*`) is stored as the value in the case of overflow data or uneditable data.

[1] This indicates the error data in microseconds of an overflow exceeding 4,200 seconds.

[2] If the HiRDB version from which the input statistics log information was obtained is older than the HiRDB version used to execute `pdstedit`, the statistics log information may not contain some of the target output items. Such an item that is not included in the statistics log information is referred to as *uneditable data*.

14.2.3 Notes on option specifications

1. A specific type of statistical information can be extracted by combining the `-k` option with other options. Table 14-2 shows the relationships between the `-k` option and the other options.

Table 14-2: Relationships between the -k option and other options

-k option (value)	**Other options**				
	-t start time and/or end time	**-u UAP name(s)**	**-x host name(s)**	**-s server name(s)**	**-f foreign -server-name**
`sys` (system activity statistical information)	Y	—	Y	—	—
`svr` (system activity statistical information by server)	Y	—	Y	Y	—
`uap` (UAP statistical information)	Y	Y	Y	—	—
`sql` (SQL statistical information)	Y	Y	Y	—	—
`buf` (global buffer pool statistical information)	Y	—	Y	Y	—
`fil` (statistical information on HiRDB files for database manipulation)	Y	—	Y	Y	—
`dfw` (deferred write processing statistical information)	Y	—	Y	Y	—
`idx` (index statistical information)	Y	—	—	Y	—
`sop` (SQL static optimization information)	Y	Y	Y	—	—
`dop` (SQL dynamic optimization information)	Y	Y	Y	—	—
`pcd` (SQL object execution information)	Y	Y	Y	—	—
`obj` (SQL object transfer statistical information)	Y	Y	Y	—	—
`sqh` (SQL statement statistical information)	Y	Y	Y	Y	—
`cnc` (`CONNECT`/`DISCONNECT` statistical information)	Y	Y	Y	Y	—

-k option (value)	Other options				
	-t start time and/or end time	-u UAP name(s)	-x host name(s)	-s server name(s)	-f foreign -server-name
`fsv` (foreign server operation statistical information)	Y	Y	Y	Y	Y
`hba` (foreign server utilization statistical information)	Y	Y	Y	Y	Y
`all` (all the above)	Y	Y	Y	Y	—

Y: Can be specified (when specified, the corresponding information can be extracted).

—: Ignored if specified (the corresponding information cannot be extracted).

2. When `-k idx` is specified, the `-o` option is ignored, if specified.
3. If the `-d` option is specified when `-k idx` or `-k all` is specified, the target information can be extracted.
4. If `sop`, `dop`, or `pcd` is specified in the `-k` option, the `-o` option must also be specified.
5. The following table shows the files that can be input and the statistical information that can be analyzed when the `-i` or `-d` option is specified:

Option	File that can be input	Statistical information that can be analyzed
`-i`	Unload log file	Information about splitting index statistical information Statistical information about `CONNECT/DISCONNECT`
	Unload statistics log file	All statistical information other than the above
`-d`	System log file	Information about splitting index statistical information Statistical information about `CONNECT/DISCONNECT`

6. When `-k uap` is specified, the statistical information acquisition units depend on the UAP execution environment. The following table shows the relationship between client environment definition (`PDSTJTRNOUT`) during UAP execution and statistical information acquisition units:

Contents of PDSTJTRNOUT	Type of UAP	
	OLTP environment	Other environment
`YES`	By transaction	By transaction
`NO`	By UAP execution	By UAP execution
Omitted	By transaction	By UAP execution

14.3 Details about statistical information

14.3.1 Output formats of statistical information

(1) Summary log record information for each edit item

Following each set of statistical information, summary log record information is output for each edit item. However, a summary log is not output for the following statistical information: SQL static optimization, SQL dynamic optimization, SQL object execution, SQL statement history, and CONNECT/DISCONNECT.

```
           :
Statistical information
           :
FILE KIND LOG KIND FIRST               LAST
XXX       XXX      XXXX/XX/XX XX:XX:XX XXXX/XX/XX XX:XX:XX
 :         :                 :                   :
       NUM
XXXXXXXXXX
     :
```

Explanation

FILE KIND

Indicates the type of log file subject to analysis.

STJ is displayed for unload statistics log files and FJ is displayed for system log files (unload system log or system log file group).

LOG KIND

Indicates the type of log record subject to analysis:

sys: System activity statistical information

uap: UAP statistical information

sql: SQL statistical information

buf: Global buffer pool statistical information

fil: Statistical information on HiRDB files for database manipulation

dfw: Deferred write processing statistical information

idx: Index statistical information

FIRST

Indicates the earliest log acquisition time for log records subject to analysis.

If the log file subject to analysis contains no log records subject to analysis, `****/**/** **:**:**` is displayed.

`LAST`

Indicates the most recent log acquisition time for log records subject to analysis.

If the log file subject to analysis contains no log records subject to analysis, `****/**/** **:**:**` is displayed.

`NUM`

Indicates the number of log records subject to analysis.

(2) Summary information on input log files

Following all statistical information, summary information on all input log files is displayed for each type of log records.

If a system log file group is input (with the `-d` option specified), summary information on input log files is not displayed. If unload log files are input, summary information on input log files is displayed only when edit items are specified for index statistical information. If any other edit items are specified, 0 is displayed as the number of records.

```
                   :
Summary log record information for each edit item
                   :
NO FILE KIND:LOG FILE NAME
 LOG KIND  FIRST               LAST                       NUM
XX XXX      :XX...X
 XXX       XXXX/XX/XX XX:XX:XX XXXX/XX/XX XX:XX:XX XXXXXXXXXX
  :                  :                    :              :
```

Explanation

`NO`

Indicates the serial number of the log file.

`FILE KIND`

Indicates the type of log file subject to analysis.

`STJ`: Unload statistics log files obtained by a HiRDB in 32-bit mode

`STJ64`: Unload statistics log files obtained by a HiRDB in 64-bit mode

`FJ`: Unload log file (regular file)

`FJIOS`: Unload log file (HiRDB file)

`LOG FILE NAME`

Indicates the name of the input unload statistics log file or unload log file (applicable to index statistical information).

The absolute pathname with a maximum length of 256 bytes is displayed as the name. If the name exceeds 256 bytes, the last 256 bytes are displayed.

`LOG KIND`

Indicates the type of log record subject to analysis:

`sys`: System activity statistical information

`uap`: UAP statistical information

`sql`: SQL statistical information

`buf`: Global buffer pool statistical information

`fil`: Statistical information on HiRDB files for database manipulation

`dfw`: Deferred write processing statistical information

`idx`: Index statistical information

`sop`: SQL static optimization information

`dop`: SQL dynamic optimization information

`pcd`: SQL object execution information

`obj`: SQL object transfer statistical information

`sqh`: SQL statement statistical information

`cnc`: `CONNECT`/`DISCONNECT` statistical information

`fsv`: Foreign server operation statistical information

`hba`: Foreign server utilization statistical information

`FIRST`

Indicates the earliest log acquisition time for the log file.

If the log file subject to analysis contains no log records subject to analysis, `****/**/** **:**:**` is displayed.

`LAST`

Indicates the most recent log acquisition time for the log file.

If the log file subject to analysis contains no log records subject to analysis, `****/**/** **:**:**` is displayed.

`NUM`

Indicates the number of records by log record type.

(3) Output information when there is no edit data

Displayed information	Description	Action
`"***** NO DATA NOOPT *****"`	Input statistics unload file subject to analysis contains no applicable edit item records.	Specified options may be invalid, or output items may be invalid when log records were acquired. Take one of the following actions: • Check the `-x`, `-s`, `-t`, `-u`, and `-f` options and correct as necessary, then re-execute. • Check the summary information on input statistics unload files to see if log records to be edited have been output and if the output time of the input statistics unload file is correct. Also, check that the correct input statistics unload files were input or that log records subject to analysis have been output.
`"***** NO DATA HOST *****"`	No log record was output from the host specified with the `-x` option.	
`"***** NO DATA SERVER *****"`	No log record was output from the server specified with the `-s` option.	
`"***** NO DATA TIME *****"`	No log record was output within the edit time specified with the `-t` option.	
`"***** NO DATA UAP *****"`	No log record was output from the UAP specified with the `-u` option.	
`"***** NO DATA FOREIGN SERVER *****"`	There is no log record for the foreign server specified with the `-f` option.	

Displayed information	Description	Action
`"***** NO DATA 1 REC *****"`	Differential information cannot be edited because the input statistics unload file subject to analysis contained no log records for acquiring differential information. The system determines that there is no log record for acquiring differential information if there is more than one log record for summary log information or for summary information in the statistics input unload file, but they were output at a single synchronization point (the earliest log acquisition time (`FIRST`) and the most recent log acquisition time (`LAST`) for the summary information are in the same period). This applies only to global buffer pool statistical information and statistical information on HiRDB files for database manipulation.	Set the unload file so that there are at least two records.
`"***** NO DATA *****"`	There was no log record subject to analysis within the specified edit period.	If `NO DATA` is displayed often for the specified edit period, increase the length of the edit period.
	There was no applicable log record for the specified edit item.	Check the summary information on input statistics unload files to see if there are editable log records.

Note

If there are no records subject to analysis for an edit item of SQL static optimization information, SQL dynamic optimization information, or SQL object execution information, an information message to that effect is displayed.

(4) Output formats of statistical information

If there are numeric values in the output information of the statistical information, they are displayed as follows:

Value range	Display
`0-999`	`0-999`
`1,000-9,994`	`1.00k-9.99k*`
`9,995-99,949`	`10.0k-99.9k*`
`99,950-999,499`	`100k-999k*`
`999,500-9,994,999`	`1.00M-9.99M*`
`9,995,000-99,949,999`	`10.0M-99.9M*`
`99,950,000-999,499,999`	`100M-999M*`
`999,500,000-9,994,999,999`	`1.00G-9.99G*`
`9,995,000,000-99,949,999,999`	`10.0G-99.9G*`
`99,950,000,000-999,499,999,999`	`100G-999G*`
`999,500,000,000-4,398,046,511,103`	`1.00T-4.40T*`

k: Kilo

M: Mega

G: Giga

T: Tera

* The last digit displayed is rounded off on the basis of the next digit.

(5) Notes about the output statistical information

To output statistical information, use the following procedure:

1. Numbers are right-justified; character strings are left-justified.
2. The output range (output dates and times) is displayed when a range was specified for editing. If no time range was specified, `*` is displayed.
3. When an edit start time was specified, the beginning editing time is for the same time zone as the edit start time. If no edit start time was specified, output begins with the applicable file record in the earliest time zone.
4. When an edit end time was specified, the ending editing time is from the applicable file record in the earliest time zone to the time zone that is the same as the edit end time.
5. If overflow occurs during summing or a cumulative value is not to be output for an item, `***`, `****`, or blanks is displayed.
6. `*` indicates data whose average value resulted in an overflow. In such a case, the

maximum and minimum values are accurate.

7. If there is no data in an edit time zone, `***** NO DATA aa...a *****` is displayed for that time zone.

8. If the unit is bytes, 1.00K = 2^{10} bytes, 1.00M = 2^{20} bytes, and 1.00G = 2^{30} bytes.

14.3.2 System activity statistical information

This section shows statistical information about the system activity.

```
pdstedit VV-RR(Object Option) ***** SYSTEM INFORMATION *****
INPUT          :/tmp/stjdata [1]
OUTPUT RANGE   :**/**/** **:**:** - **/**/** **:**:** [2]
-----------------------------------------------------------------------------
HOST = test [3]
-----------------------------------------------------------------------------
EDIT TIME 2000/12/11 14:00:00 - 2000/12/11 14:20:00 [4]

SERVER : ******** [5]                        [6]   [7]   [8]   [9]
                                             FREQ  MAX   MIN   AVG
                                             ----- ----- ----- -----
<SCHEDULE>[10]     QUEUE LENGTH[11]             66     1     1     1
                   MESSAGE LENGTH[12]           66 1.55k   548   973

<PROCESS>[13]      # OF USER SERVER ABORT[14]    0
                   # OF SYSTEM SERVER ABORT[15]  0
                   # OF PROCESS[16]                   13    10    12
                   # OF PROCESS ON SERVICE[17]         3     0     0
                   # OF REQ PROCESS OVER MAX[18] 0

<TRANSACTION>[19]  # OF COMMIT[20]             105
                   # OF ROLLBACK[21]             3
<NAME>[22]         # OF CACHE HIT[23]            0
                   # OF LOCAL HIT[24]          364
                   # OF LOOK-UP[25]            364

<RPC>[26]          # OF TIME OUT[27]             0
                   # OF ERROR[28]                0
                   RESPONSE ON OWN UNIT[29]     81  308k     1 10.2k
                   RESPONSE TO OTHER UNIT[30]    0     0     0     0
                   EXEC TIME ON OWN UNIT[31]   759  311k     0 2.19k
                   EXEC TIME FROM OTHER UNIT[32] 0     0     0     0
                   # OF SEND TO OWN PRCS[33]     0

                   # OF SEND TO OTHER PRCS[34]  63
                   # OF SEND TO OTHER UNIT[35]   0
                   # OF RECV FROM OWN PRCS[36]   0
                   # OF RECV FROM OTHER PRCS[37] 0
                   # OF RECV FROM OTHER UNIT[38] 0
                   # OF REGISTERED PORTS[39]     0     0     0     0
                   # OF ASSIGNED PORTS[40]       0     0     0     0
```

```
<LOCK>[41]         WAIT TIME[42]                 0     0     0     0
                   QUEUE LENGTH[43]              0     0     0     0
                   # OF DEADLOCK[44]             0
                   % OF USE LOCK TABLE[45]       1     0     0     0

<SHARED MEMORY>[46]STATIC GET SIZE[47]           0     0     0     0
                   STATIC POOL SIZE[48]          0     0     0     0
                   DYNAMIC GET SIZE[49]        216 1.20M 1.19M 1.19M
                   DYNAMIC POOL SIZE[50]       216  657k  652k  655k
                   SIZE EXCEPT GLOBAL BUFFER[51]                 23.3M
                   STATIC SIZE[52]                               1.58M
                   DYNAMIC SIZE[53]                              1.19M
                   SIZE FOR GLOBAL BUFFER[54]                     344k

<SYNC POINT>[55]   SYNC POINT GET INTERVAL[56]   2  109k 44.4k 76.8k
                   SYNC POINT GET TIME[57]       2 1.01k 1.01k 1.01k
<LOG>[58]          # OF BUFFER FULL[59]         35
                   # OF WAIT THREAD[60]          0
                   OUTPUT BLOCK LENGTH[61]     418 32.0k   332 8.26k
                   NOT BUS LENGTH[62]          418 32.0k   332 8.26k
                   # OF BUFFER FOR WAIT I/O[63]  0     0     0     0
                   # OF WRITE TO FILE[64]      427
                   # OF WRITE ERROR[65]          0
                   LOG FILE SWAP TIME[66]        3   131    60    90
                   LOG INPUT DATA LENGTH[67]    96 32.0k 32.0k 32.0k
                   # OF READ FROM FILE[68]      99
                   # OF READ ERROR[69]           0

<DICTIONARY>[70]   # OF TBL-DEF GET REQ[71]     29
                   # OF TBL-CACHE HIT[72]       28
                   # OF CACHED TBL-DEF[73]       1     7     7     7
                   USED TBL-DEF SIZE[74]         1 9.23k 9.23k 9.23k
                   TBL-CACHE SIZE[75]            1 60.2k 60.2k 60.2k
                   # OF ACCESS PRIV CHECK[76]    0
                   # OF CACHE HIT (AP CHECK)[77] 0
                   # OF CON/DBA DEF GET REQ[78] 39
                   # OF CON/DBA CACHE HIT[79]   39

                   # OF CON/DBA CACHED USER[80]  0     0     0     0
                   DICT SERV TRANS DATA SIZE[81] 0     0     0     0
                   # OF TRANS[82]                0
                   # OF VIEW DEF GET REQ[83]     0
                   # OF VIEW CACHE HIT[84]       0
                   # OF VIEW CACHED DEF[85]      0
                   USED VIEW SIZE[86]            0     0     0     0
                   VIEW CACHE SIZE[87]           0     0     0     0
                   CACHE-MISS VIEW SIZE[88]      0     0     0     0

                   # OF TYPE-DEF GET REQ[89]     0
                   # OF TYPE-DEF CACHE HIT[90]   0
                   # OF CACHED TYPE-DEF[91]      0     0     0     0
                   TYPE-DEF CACHE SIZE[92]       0     0     0     0
                   TYPE-DEF CACHE TOTAL SIZE[93] 0     0     0     0
                   TYPE-DEF CACHE ALLOC SIZE[94] 0     0     0     0
                   # OF RTN-DEF GET REQ[95]      0
                   # OF RTN-DEF CACHE HIT[96]    0
                   # OF CACHED RTN-DEF[97]       0     0     0     0
                   RTN-DEF CACHE SIZE[98]        0     0     0     0
                   RTN-DEF CACHE TOTAL SIZE[99]  0     0     0     0
                   RTN-DEF CACHE ALLOC SIZE[100] 0     0     0     0

                   # OF PLG-RTN GET REQ[101]     0
                   # OF PLG-RTN CACHE HIT[102]   0
                   # OF REGISTRY GET REQ[103]    0
                   # OF REGISTRY CACHE HIT[104]  0
                   # OF CACHED REGISTRY-DEF[105] 0     0     0     0
                   REGISTRY CACHE SIZE[106]      0     0     0     0
                   REGISTRY CACHE TOTAL SIZE[107]0     0     0     0
                   DIRECTORY USER CHECK TIME[108]2 33.0k 20.2k 26.6k
                   GROUP CHECK TIME[109]         2   217   179   198

<FES-BES-DIC(SDS)  # OF SQLOBJ INFO GET[111]    90
 INFORMATION>      # OF CACHE HIT (SQLOBJ)[112] 63
 [110]             # OF CACHED SQLOBJ[113]      27    32     6    19
                   CACHED SQLOBJ TOTAL SIZE[114]27 86.0k 17.0k 52.0k
                   # OF SWAP OUT SQLOBJ[115]     0
                   REQUEST SQLOBJ SIZE[116]     27 3.65k 1.74k 2.67k
                   # OF STRT INFO GET[117]       0
                   # OF CACHED HIT (STRT)[118]   0

                   # OF CACHED STRT[119]         0     0     0     0
                   CACHED STRT TOTAL SIZE[120]   0     0     0     0
                   # OF SWAP OUT STRT[121]       0
                   REQUEST STRT SIZE[122]        0     0     0     0
                   # OF STRT RECOMPILE[123]      0
-----------------------------------------------------------------------------
FILE KIND   LOG KIND   FIRST                 LAST                        NUM
STJ        sys        2000/12/11 14:08:29   2000/12/11 14:08:29           1
-----------------------------------------------------------------------------


NO FILE KIND:LOG FILE NAME
  LOG KIND             FIRST                 LAST                        NUM
 1 STJ      :/tmp/stjdata/pdstj01
  sys                  ****/**/** **:**:**   ****/**/** **:**:**           0
  uap                  2000/12/11 15:33:53   2000/12/11 15:36:53         104
  sql                  2000/12/11 15:33:53   2000/12/11 15:36:53         536
  sop                  2000/12/11 15:33:53   2000/12/11 15:34:05          12
  dop                  2000/12/11 15:33:53   2000/12/11 15:36:53         144

  pcd                  ****/**/** **:**:**   ****/**/** **:**:**           0
  obj                  ****/**/** **:**:**   ****/**/** **:**:**           0
  sqh                  ****/**/** **:**:**   ****/**/** **:**:**           0
  buf                  2000/12/11 15:33:41   2000/12/11 15:36:54          18
  fil                  2000/12/11 15:33:41   2000/12/11 15:36:54         150
  dfw                  2000/12/11 15:33:41   2000/12/11 15:36:54           7
  idx                  2000/12/11 15:33:41   2000/12/11 15:36:54           6
 2 STJ      :/tmp/stjdata/pdstj02
  sys                2000/12/11 14:08:29   2000/12/11 14:08:29            2
  uap                2000/12/11 14:06:30   2000/12/11 14:09:07           52
  sql                2000/12/11 14:06:30   2000/12/11 14:09:07          496
  sop                2000/12/11 14:06:30   2000/12/11 14:07:14           27
  dop                2000/12/11 14:06:30   2000/12/11 14:09:06          120
  pcd                ****/**/** **:**:**   ****/**/** **:**:**            0

  obj                ****/**/** **:**:**   ****/**/** **:**:**            0
  sqh                ****/**/** **:**:**   ****/**/** **:**:**            0
  buf                2000/12/11 14:06:30   2000/12/11 14:09:08           12
  fil                2000/12/11 14:06:30   2000/12/11 14:09:08          100
  dfw                2000/12/11 14:06:30   2000/12/11 14:09:08         1077
  idx                2000/12/11 14:06:30   2000/12/11 14:09:08            4

 3 STJ      :/tmp/stjdata/pdstj03
  sys                ****/**/** **:**:**   ****/**/** **:**:**            0
  uap                2000/12/11 15:33:53   2000/12/11 15:36:53          104
  sql                2000/12/11 15:33:53   2000/12/11 15:36:53          536
  sop                2000/12/11 15:33:53   2000/12/11 15:34:05           12
  dop                2000/12/11 15:33:53   2000/12/11 15:36:53          144

  pcd                ****/**/** **:**:**   ****/**/** **:**:**            0
  obj                ****/**/** **:**:**   ****/**/** **:**:**            0
  sqh                ****/**/** **:**:**   ****/**/** **:**:**            0
  buf                2000/12/11 15:33:41   2000/12/11 15:36:54           18
  fil                2000/12/11 15:33:41   2000/12/11 15:36:54          150
  dfw                2000/12/11 15:33:41   2000/12/11 15:36:54            7
  idx                2000/12/11 15:33:41   2000/12/11 15:36:54            6

 4 STJ      :/tmp/stjdata/pdstj04
  sys                ****/**/** **:**:**   ****/**/** **:**:**            0
  uap                2000/12/11 15:58:46   2000/12/11 15:59:18           52
  sql                2000/12/11 15:58:46   2000/12/11 15:59:18          268
  sop                2000/12/11 15:58:45   2000/12/11 15:58:55           17
  dop                2000/12/11 15:58:45   2000/12/11 15:59:17           72
  pcd                ****/**/** **:**:**   ****/**/** **:**:**            0

  obj                ****/**/** **:**:**   ****/**/** **:**:**            0
  sqh                ****/**/** **:**:**   ****/**/** **:**:**            0
  buf                2000/12/11 15:58:39   2000/12/11 15:59:18            9
  fil                2000/12/11 15:58:39   2000/12/11 15:59:18           75
  dfw                2000/12/11 15:58:39   2000/12/11 15:59:18            3
  idx                2000/12/11 15:59:10   2000/12/11 15:59:18            2
```

Explanation

1. Name of input statistics unload file or name of the directory containing the input statistics unload file (maximum of 58 bytes)
2. Output range (output start date/time to output end date/time)
3. Name of host requesting output of system activity statistical information
4. Edit period (collection start time to collection end time)

5. Server name. If `sys` was specified in the `-k` option, `********` is displayed
6. Number of occurrences of each item
7. Maximum values
8. Minimum values
9. Average values
10. Schedule information
11. Number of requests in the schedule queue

    This information includes the number of requests to the server registered in the schedule queue and the maximum, minimum, and average numbers of those requests in the schedule queue at any one time.
12. Length of schedule message (bytes)

    This is the number of messages containing the server processing requests and the maximum, minimum, and average message lengths.
13. Process information
14. Number of times the server terminated abnormally

    This is the number of times the single server, front-end server, dictionary server, and back-end server processes terminated abnormally.
15. Number of times the HiRDB internal server terminated abnormally

    This is the number of times the HiRDB's internal server processes terminated abnormally.
16. Number of internal server processes that were used by servers and HiRDB

    This is the number of internal server processes used by a single server, front-end server, dictionary server, back-end server, and HiRDB.
17. Number of server processes in service

    This is the number of server processes currently providing services.
18. Number of service requests exceeding the maximum number of startup processes

    This is the number of service requests that exceeds the maximum number of processes that can be started. For the single server or front-end server, the maximum number of startup processes is the value of the `pd_max_users` operand; for the back-end server, it is the value of the `pd_max_bes_process` operand; for the dictionary server, it is the value of the `pd_max_dic_process` operand. The number of service requests may be greater than the actual number of excess requests because it includes the retried requests.
19. Transaction information

20. Commits count

    This is the number of transactions that committed among all the transactions processed at the single server, front-end server, dictionary server, and back-end server.

21. Rollbacks count

    This is the number of transactions that rolled back among all the transactions processed at the single server, front-end server, dictionary server, and back-end server.

22. Name server information
23. Information used by the system (not by users)
24. Information used by the system (not by users)
25. Information used by the system (not by users)
26. RPC information
27. Information used by the system (not by users)
28. Information used by the system (not by users)
29. Service response time for local unit's servers (in 100 microseconds)

    This information includes the number of service responses for the local unit's servers and the maximum, minimum, and average response time.

30. Service response time for remote unit's servers (in 100 microseconds)

    This information includes the number of service responses for the remote unit's servers and the maximum, minimum, and average response time.

31. Execution time per service from local unit's servers (in 100 microseconds)

    This information includes the number of executions per service from the local unit's servers and the maximum, minimum, and average execution time.

32. Execution time per service from remote unit's servers (in 100 microseconds)

    This information includes the number of executions per service from the remote unit's servers and the maximum, minimum, and average execution time.

33. Number of `SEND`s to local process

    This is the number of times `SEND` (message send) was executed on the local process.

34. Number of `SEND`s to other processes on local unit

    This is the number of times `SEND` (message send) was executed on other processes on the local unit.

35. Number of `SEND`s to remote unit

    This is the number of times `SEND` (message send) was executed on the remote unit.

36. Number of `RECEIVE`s from local unit

    This is the number of times `RECEIVE` (message receive) was executed from the local unit.

37. Number of `RECEIVE`s from other processes on local unit

    This is the number of times `RECEIVE` (message receive) was executed from other processes on the local unit.

38. Number of `RECEIVE`s from remote unit

    This is the number of times `RECEIVE` (message receive) was executed from the remote unit.

39. Number of HiRDB-reserved ports used

    This is the number of port numbers actually used among all the port numbers reserved by HiRDB, and the maximum, minimum, and average values of port numbers.

40. Number of additional ports automatically allocated by the OS when the system ran out of ports reserved by HiRDB

    This is the number of ports automatically allocated by OS when all the port numbers reserved by HiRDB were in use, and the maximum, minimum, and average values of port numbers.

41. Lock information

42. Lock release wait time (milliseconds)

    This information includes the number of lock requests in the server that resulted in the lock release wait status because the requested resource was already locked by another user, and the maximum, minimum, and average wait time.

43. Number of users placed in lock release wait status

    This information includes the number of users in the server that resulted in the lock release wait status because the requested resource was already locked by another user, and the maximum, minimum, and average numbers of the users placed in lock release wait status.

44. Deadlocks count

    This is the number of times a lock request resulted in deadlock in the server.

45. Utilization factor of locked resources management table (%)

This information includes the number of events that increased the utilization factor of the locked resources management table by 5% in the server, and the maximum, minimum, and average utilization factors.

46. Shared memory information
47. Information used by the system (not by users)
48. Information used by the system (not by users)
49. Information used by the system (not by users)
50. Information used by the system (not by users)
51. Size of shared memory allocated for the server and HiRDB internal server (bytes)

    This is the size of shared memory allocated on the unit for the server and HiRDB internal server.
52. Size of shared static memory (bytes)

    Of the memory allocated on the unit for the server or HiRDB internal server, this is the size allocated as the shared static memory.
53. Size of shared dynamic memory (bytes)

    Of the memory allocated on the unit for the server or HiRDB internal server, this is the size allocated as the shared dynamic memory.
54. Allocation size of shared memory for global buffer pool (bytes)

    This is the size of the global buffer allocated in the server (if the global buffer uses multiple shared memory segments, this is the total size).
55. Synchronization point information
56. Synchronization point dump interval (milliseconds)

    This information includes the number of times a synchronization point dump is collected and the maximum, minimum, and average intervals between two synchronization point dump collections.
57. Synchronization point dump time (milliseconds)

    This information includes the number of synchronization point dumps and the maximum, minimum, and average time required for collecting a synchronization point dump. If this value is large, it may have taken a long time to validate the synchronization point dump because a transaction requiring a long processing time was executed concurrently.
58. Log information
59. Buffer-fulls count

    This is the number of times the log output buffer became full at the HiRDB system or server.

60. Waits count due to a shortage of current buffer

    This is the number of times threads were placed in wait status during output operation on the log input/output buffer at the HiRDB system or server because the log input/output buffer was being used for output operation on the system log file.

61. Output block length (bytes)

    This information includes the number of times data was output from the log input/output buffer to the system log file at the HiRDB system or server and the maximum, minimum, and average block lengths.

    For bus output, this is the total length. If a file output request is issued for a log buffer while file output is underway on another log buffer, HiRDB keeps storing the remaining system log in the log buffer requested for file output until the current log buffer being output is released. When the current log buffer is released, HiRDB then outputs both at the same time. This is called bus output.

62. Data length excluding the bus output (bytes)

    This information includes the number of times data was output from the log input/output buffer to system log file, excluding the bus output, at the HiRDB system or server, and the maximum, minimum, and average log block lengths.

63. Number of buffer sectors placed in output wait status

    This information includes the number of log output buffer sectors waiting for completion of output to the system log file when output to the system log file was completed at the HiRDB system or server, and the maximum number of such buffer sectors × 100, the minimum number of such buffer sectors × 100, and the average number of such buffer sectors × 100.

64. Number of times data was written into file

    This is the number of times data was written into system log file at the HiRDB system or server.

    This value includes the number of write operations executed to change the status of system log file. If dual system log files are used, the value also includes the number of write operations for each file version. Additionally, this value includes the number of system log file output operations that occur when the file status changes due to file swapping.

65. Number of times write error occurred

    This is the number of times a system log file write error occurred at the HiRDB system or server.

66. Log file swapping time (milliseconds)

This information includes the number of times swapping occurred on the system log files at the HiRDB system or server, and the maximum, minimum, and average time elapsed before system log files were swapped.

67. Length of log input data (bytes)

    This information includes the number of logged entries made from the system log file during rollback at the HiRDB system or server, and the maximum, minimum, and average lengths of logged data.

68. File reads count

    This is the number of times logged data was read from the system log file during rollback at the HiRDB system or server.

69. Number of read errors

    This is the number of times a system log file read error occurred at the HiRDB system or server.

70. Dictionary information

71. Number of table definition information acquisition requests

    This is the number of times table manipulation was executed in the single server or front-end server.

72. Table definition information buffer hits count

    This is the number of times requested table definition information was found in the table definition information buffer in the single server or front-end server.

    If the table definition information buffer hits count is less than the number of table definition information acquisition requests, the size of the table definition information buffer should be reevaluated.

73. Number of table definition information items in the table definition information buffer

    This is the number of table definition information items buffered in the table definition information buffer in the single server or front-end server, and the maximum, minimum, and average numbers of table definition information items.

74. Length of table definition information placed in the table definition information buffer (bytes)

    This information includes the number of table definition information items placed in the table definition information buffer in the single server or front-end server, and the maximum, minimum, and average sizes of the table definition information buffer used per table definition information item.

75. Size of table definition information buffer used (bytes)

This information includes the number of table definition information items placed in the table definition information buffer in the single server or front-end server, and the maximum, minimum, and average values of the total length of the table definition information.

If the maximum value is extremely small compared to the size of the table definition information buffer, the table definition information buffer may be too large. In this case, you should reevaluate the size of this buffer.

76. Table access privilege information acquisitions count

    This is the number of times table access privilege was acquired to manipulate a table at the single server or front-end server.

77. Table access privilege information buffer hits count

    This is the number of times the requested table access privilege was found in the table access privilege information buffer at the single server or front-end server.

    If this value is extremely small compared to the acquisitions count, privileges may be granted to more than 100 users per table. In this case, you should evaluate whether `PUBLIC` privileges can be granted for the corresponding table.

78. Number of user privilege information acquisition requests

    This is the number of times a `CONNECT` request was issued from a UAP to a single server or front-end server.

79. User privilege information buffer hits count

    This is the number of times the requested user privilege information was found in the user privilege information buffer for the `CONNECT` requests issued from a UAP to a single server or front-end server.

    If this value is less than the number of user privilege information acquisition requests, you should reevaluate the size of the user privilege information buffer.

80. Number of user privilege information buffer users

    This information includes the number of times the requested user privilege information was placed in the user privilege information buffer for the `CONNECT` requests issued from a UAP to a single server or front-end server, and the maximum, minimum, and average numbers of such users.

    If the maximum value times 34 (bytes) is extremely small compared to the size of the user privilege information buffer, you should reevaluate the size of the user privilege information buffer.

81. Length of data handled during communication with dictionary server (bytes)

    This information includes the number of table definition information acquisition requests when the requested table definition information was not found in the table definition information buffer at the front-end server, and the maximum, minimum, and average lengths of data handled during communication with the dictionary server.

82. Dictionary server communications count

    This is the number of times communication was established with the dictionary server to acquire table definition information because the requested table definition information was not found in the table definition information buffer at the front-end server.

83. Number of view analysis information acquisition requests

    This is the number of times view analysis information was acquired to manipulate a view at the single server or front-end server.

84. View analysis information buffer hits count

    This is the number of times the requested view analysis information was found in the view analysis information buffer at the single server or front-end server.

    If the view analysis information buffer hits count is less than the number of view analysis information acquisition requests, you should reevaluate the size of the view analysis information buffer.

85. Number of analysis information items in view analysis information buffer

    This is the number of view analysis information items placed in the view analysis information buffer.

86. Size of view analysis information buffer used per view definition information item (bytes)

    This information includes the number of view analysis information items placed in the view analysis information buffer, and the maximum, minimum, and average sizes of the view analysis information buffer used per view analysis information item.

87. Size of view analysis information buffer used (bytes)

    This information includes the number of view analysis information items in the view analysis information buffer, and the maximum, minimum, and average sizes.

    If the maximum value is extremely small compared to the size of the view analysis information buffer, the view analysis information buffer may be too large. In this case, you should evaluate whether the size of the view analysis information buffer is reasonable.

88. Size of view analysis information resulting in buffer mist (bytes)

    This information includes the number of times the requested view analysis information was retrieved from the dictionary because it was not found in the view analysis information buffer at the single server or front-end server, and the maximum, minimum, and average sizes of such view analysis information.

If the total value is close to the size of the view analysis information buffer, you should increase the size of the view analysis information buffer.

89. Number of type definition information acquisition requests

    This is the number of times type definition was referenced at the single server or front-end server.

90. User-defined type information buffer hits count

    This is the number of times the requested type definition information was found in the user-defined type information buffer at the single server or front-end server.

91. Number of type definition information items in the user-defined type information buffer

    This is the number of table definition information items buffered in the user-defined type information buffer in the single server or front-end server, and the maximum, minimum, and average numbers of type definition information items.

92. Size of user-defined type information buffer used per type definition information item (bytes)

    This information includes the number of type definition information items placed in the user-defined type information buffer at the single server or front-end server, and the maximum, minimum, and average sizes of the user-defined type information buffer used per type definition information item.

93. Total size of user-defined type information buffer used (bytes)

    This information includes the number of type definition information items acquired at the single server or front-end server, and the maximum, minimum, and average sizes of the user-defined type information buffer.

    If the maximum value is less than the size of the allocated user-defined type information buffer, the buffer size may be too large. You should reevaluate the size of the user-defined type information buffer.

94. Size of allocated user-defined type information buffer (bytes)

    This information includes the number of buffer sectors allocated for user-defined type information at the single server or front-end server, and the maximum, minimum, and average sizes of the user-defined type information buffer. In this case, the maximum, minimum, and average values are the same.

95. Number of routine definition information acquisition requests

This is the number of times routine definition was referenced at the single server or front-end server.

96. Routine definition information buffer hits count

    This is the number of times the requested routine definition information was found in the routine definition information buffer at the single server or front-end server.

    If the routine definition information buffer hits count is less than the number of routine definition information acquisition requests, you should reevaluate the size of the routine definition information buffer.

97. Number of routine definition information items in the routine definition information buffer

    This is the number of routine definition information items buffered in the routine definition information buffer in the single server or front-end server, and the maximum, minimum, and average numbers of routine definition information items.

98. Size of routine definition information buffer used per routine definition information item that was placed in the routine definition information buffer (bytes)

    This information includes the number of routine definition information items placed in the routine definition information buffer at the single server or front-end server, and the maximum, minimum, and average sizes of the routine definition information buffer per routine definition information item.

99. Total size of routine definition information buffer (bytes)

    This information includes the number of items placed in the routine definition information buffer at the single server or front-end server, and the maximum, minimum, and average sizes of the routine definition information buffer used.

100. Size of allocated routine definition information buffer (bytes)

    This information includes the number of buffer sectors allocated for routine definition information at the single server and front-end server, and the maximum, minimum, and average sizes of the routine definition information buffer. In this case, the maximum, minimum, and average values are the same.

101. Number of routine definition acquisition requests for plug-in function

    This is the number of times a plug-in function's routine was referenced at the single server or front-end server.

102. Plug-in function's routine definition information buffer hits count

    This is the number of times a requested plug-in function's routine information was found in the routine definition information buffer at the single server or front-end server.

If the plug-in function's routine definition information buffer hits count is less than the number of routine definition acquisition requests for the plug-in function, you should reevaluate the size of the routine definition information buffer.

103. Number of registry information acquisition requests

     This is the number of times registry information was referenced at the single server or front-end server.

104. Registry information buffer hits count

     This is the number of times the requested registry information was found in the registry information buffer at the single server or front-end server.

     If the registry information buffer hits count is less than the number of registry information acquisition requests, you should reevaluate the size of the registry information buffer.

105. Number of registry information items in the registry information buffer

     This is the number of registry information items buffered in the registry information buffer in the single server or front-end server, and the maximum, minimum, and average numbers of registry information items.

106. Size of registry information buffer per registry information item (bytes)

     This information includes the number of registry information items placed in the registry information buffer at the single server or front-end server, and the maximum, minimum, and average sizes of the registry information buffer per registry information item.

107. Total size of registry information buffer used (bytes)

     This information includes the number of registry information items placed in the registry information buffer at the single server or front-end server, and the maximum, minimum, and average sizes of the registry information buffer used.

108. User authentication time for directory registration (microseconds)

     This information includes the number of user authentication requests issued to determine whether the user is registered in the Directory Server, and the maximum, minimum, and average execution time.

109. Group checking time (microseconds)

     This information includes the number of times checking was requested to determine whether the user belongs to a Directory Server role, and to determine the maximum, minimum, and average execution time.

110. SQL object information for single server, front-end server, back-end server, or dictionary server

111. Number of SQL object acquisition requests

    This is the number of times an SQL object was acquired at the single server, front-end server, back-end server, or dictionary server.

112. SQL object buffer hits count

    This is the number of times a requested SQL object was found in the SQL object buffer at the single server, front-end server, back-end server, or dictionary server.

    If the SQL object buffer hits count is less than the number of SQL object acquisition requests, you should reevaluate the size of the SQL object buffer.

113. Number of SQL objects in the SQL object buffer

    This information includes the number of SQL objects in the SQL object buffer at the single server, front-end server, back-end server, and dictionary server, the information update count for the number of SQL objects, and the maximum, minimum, and average values. The information update count is the number of times one of the maximum, minimum, or average number of SQL objects was updated.

114. Total length of SQL objects in the SQL object buffer (bytes)

    This information includes the information update count for the total size of SQL objects in the SQL object buffer at the single server, front-end server, back-end server, and dictionary server, and the maximum, minimum, and average values. The information update count is the number of times one of the maximum, minimum, or average value of this total size of SQL objects was updated.

    Note

    HiRDB manages the SQL object buffer in increments of 1 kilobyte. Therefore, each SQL object uses an SQL object buffer whose size is at least the size of the SQL object rounded up to a full kilobyte. The sum of these SQL object sizes is not the total size of the space used by the SQL objects. There may not be free space in the SQL object buffer even when the sum of the SQL object sizes is less than the value specified as the size of the SQL object buffer.

115. Number of SQL objects taken out of the SQL object buffer

    This is the number of SQL objects that were invalidated in the SQL object buffer to make space for new SQL objects at the single server, front-end server, back-end server, or dictionary server.

116. Length of SQL object (bytes)

    This information includes the number of SQL objects placed in the SQL object buffer at the single server, front-end server, back-end server, or dictionary server, and the maximum, minimum, and average lengths of the SQL object.

117. Number of stored procedure object acquisition requests

    This is the number of requests issued to place stored procedure objects in the SQL object buffer at the single server or front-end server.

    For the back-end server and dictionary server, this value is included in the number of SQL object acquisition requests.

118. SQL object buffer hits count for stored procedure objects

    This is the number of times a requested stored procedure object was found in the SQL object buffer at the single server or front-end server.

    If the SQL object buffer hits count for stored procedure objects is less than the number of stored procedure object acquisition requests at the single server or front-end server, you should reevaluate the size of the SQL object buffer.

    For the back-end server and dictionary server, this value is included in the SQL object buffer hits count.

119. Number of stored procedure objects placed in the SQL object buffer

    This is the number of stored procedure objects buffered in the SQL object buffer in the single server or front-end server, and the maximum, minimum, and average numbers of objects.

    For the back-end server and dictionary server, this value is included in the number of SQL objects in the SQL object buffer.

120. Total length of stored procedure objects in the SQL object buffer

    This information includes the number of times a stored procedure object was acquired from the SQL object buffer at the single server or front-end server, and the maximum, minimum, and average lengths of such objects.

    For the back-end server and dictionary server, this value is included in the total length of SQL objects in the SQL object buffer.

121. Number of stored procedure objects taken out of the SQL object buffer

    This is the number of stored procedure objects invalidated in the SQL object buffer to create space for new stored procedure objects at the single server or front-end server.

    For the back-end server and dictionary server, this value is included in the number of SQL objects taken out of the SQL object buffer.

122. Length of stored procedure objects (bytes)

    This information includes the number of stored procedure objects placed in the SQL object buffer at the single server or front-end server, and the maximum, minimum, and average lengths of such objects.

For the back-end server and dictionary server, this value is included in the length of SQL objects.

123. Number of times stored procedure objects were recompiled

This is the number of times stored procedure objects were recompiled due to changes made to indexes at the single server or front-end server.

This information is not applicable to the back-end server or dictionary server.

14.3.3 UAP statistical information

The following shows the statistical information about UAPs.

```
pdstedit VV-RR(Object Option) ***** UAP INFORMATION *****
INPUT          :/tmp/stjdata [1]
OUTPUT RANGE   :**/**/** **:**:** - **/**/** **:**:** [2]
-----------------------------------------------------------------------------
HOST = test [3]
-----------------------------------------------------------------------------
EDIT TIME 2003/04/03 22:00:00 - 2003/04/03 23:00:00 [4]

UAP NAME [5]
  SERVICE NAME [6]               [7]   [8]   [9]   [10]  [11]  [12]  [13]
                                 EXEC  NORM  ERROR TTIME TAVG  CACHE LOCK
------------------------------   ----- ----- ----- ----- ----- ----- -----
                                 [14]  [15]  [16]  [17]  [18]  [19]
                                 LMAX  LAVG  CTIME CTAVG SVEXT SVEXA
                                 ----- ----- ----- ----- -----
                                 [20]  [21]  [22]  [23]  [24]  [25]  [26]
   *SQL OBJECT*                  REQ   RAVG  HITS  HAVG  CRT   CAVG  SMAX
                                 ----- ----- ----- ----- ----- ----- -----

         [27]  [28]  [29]  [30]  [31]  [32]  [33]  [34]  [35]  [36]  [37]
   *LN1* COMT  ROLB  FROW  DROW  IROW  UROW  SET   OPEN  FETC  CLOS  DESC
         ----- ----- ----- ----- ----- ----- ----- ----- ----- ----- -----
         [38]  [39]  [40]  [41]  [42]  [43]  [44]  [45]  [46]  [47]  [48]
   *LN2* SEL   INS   UPD   DEL   LOCK  CRTT  DRPT  ALTT  CRTI  DRPI  CMTT
         ----- ----- ----- ----- ----- ----- ----- ----- ----- ----- -----
         [49]  [50]  [51]  [52]  [53]  [54]  [55]  [56]  [57]  [58]  [59]
   *LN3* CMTC  CRTS  DRPS  GRTR  GRTS  GRTA  GRTC  GRTD  RVKR  RVKS  RVKA
         ----- ----- ----- ----- ----- ----- ----- ----- ----- ----- -----
         [60]  [61]  [62]  [63]  [64]  [65]  [66]  [67]  [68]  [69]  [70]
   *LN4* RVKC  RVKD  CRTV  DRPV  PRGT  CRTP  DRPP  ALTP  CALL  DESI  MISC
         ----- ----- ----- ----- ----- ----- ----- ----- ----- ----- -----

                                 [71]  [72]  [73]  [74]
   *STORED ROUTINE OBJECT*       REQ   RAVG  HITS  HAVG
                                 ----- ----- ----- -----
                                 [75]  [76]  [77]  [78]  [79]
   *DB ACCESS*                   MAXIO MINIO MWFN  MWFEC MWFVL
                                 ----- ----- ----- ----- -----
                                 [80]  [81]  [82]  [83]
                                 MWHTS MBSL1 MBSL2 MBSL3
                                 ----- ----- ----- -----
         [84]  [85]  [86]  [87]  [88]  [89]  [90]  [91]  [92]  [93]  [94]
   *LN1* IOTIM DIDRC DIDUC DIDHC DIDHR DIDRD DIDWT LBRFC LBUPC LBRHC LBUHC
         ----- ----- ----- ----- ----- ----- ----- ----- ----- ----- -----
         [95]  [96]  [97]  [98]  [99]  [100] [101] [102] [103] [104] [105]
   *LN2* LRFHR LUPHR LBRDC LBWTC BFSHC BRDWC BWTWC BLKWC WFRDC WFWTC WBFOC
         ----- ----- ----- ----- ----- ----- ----- ----- ----- ----- -----
         [106] [107] [108] [109] [110] [111] [112] [113]  [114] [115] [116]
   *LN3* SCHSK SCHCH LDIRC LDIUC LDIHC LDIRD LDIWT LBFSHC ARREQ ARWC  ARWT
         ----- ----- ----- ----- ----- ----- ----- -----  ----- ----- -----
         [117] [118] [119]
   *LN4* ARSTA HJMAX HJAVG
         -----

uap01
  service01
                                    28    28     0 6.29k   225 3.93k     0
                                     0     0 1.06k    38 5.85k   209
   *SQL OBJECT*                    136     5    37     1    99     4 4.13k
   *LN1*   123     0   503     0     0     0   136   136   526   119     0 [118]
      **     4     0    18     0     0     0     5     5    19     4     0 [119]

   *LN2*     0     0     0     0     0     0     0     0     0     0     0
      **     0     0     0     0     0     0     0     0     0     0     0
   *LN3*     0     0     0     0     0     0     0     0     0     0     0
      **     0     0     0     0     0     0     0     0     0     0     0
   *LN4*     0     0     0     0     0     0     0     0     0     0     0
      **     0     0     0     0     0     0     0     0     0     0     0

   *STORED ROUTINE OBJECT*           0     0     0     0
   *DB ACCESS*                       0     0     0     0     0
                                     0     0     0     0
   *LN1*    12 1.37k     0 1.21k    89   156     0     0     0     0     0
      **     0    49     0    43           6     0     0     0     0     0

   *LN2*     0     0     0     0   141     0     0     0     0     0     0
      **                 0     0     5     0     0     0     0     0     0
   *LN3*     0     0     0     0     0     0     0     0      0     0     0
      **                 0     0     0     0     0     0      0     0     0
   *LN4*     0     0     0
      **     0     0     0

uap02
  service02
                                     5     5     0 5.28k 1.06k 6.77k     0
                                     0     0   820   164 5.03k 1.01k
   *SQL OBJECT*                    106    21    29     6    77    15 4.13k
   *LN1*   110     0   490     0     0     0   106   106   496   106     0
      **    22     0    98     0     0     0    21    21    99    21     0

   *LN2*     0     0     0     0     0     0     0     0     0     0     0
      **     0     0     0     0     0     0     0     0     0     0     0
   *LN3*     0     0     0     0     0     0     0     0     0     0     0
      **     0     0     0     0     0     0     0     0     0     0     0
   *LN4*     0     0     0     0     0     0     0     0     0     0     0
      **     0     0     0     0     0     0     0     0     0     0     0
   *STORED ROUTINE OBJECT*           0     0     0     0
   *DB ACCESS*                       0     0     0     0     0
                                     0     0     0     0
   *LN1*    14 1.30k     0 1.12k    87   175     0     0     0     0     0
      **     3   259     0   224          35     0     0     0     0     0

   *LN2*     0     0     0     0   137     0     0     0     0     0     0
      **                 0     0    27     0     0     0     0     0     0
   *LN3*     0     0     0     0     0     0     0     0      0     0     0
      **                 0     0     0     0     0     0      0     0     0
   *LN4*     0     0     0
      **     0     0     0

*TOTAL* [120]                       33    33     0 11.6k   351 4.36k     0
                                     0     0 1.88k    57 10.9k   330
   *SQL OBJECT*                    242     7    66     2   176     5 4.13k
   *LN1*   233     0   993     0     0     0   242   242 1.02k   225     0
      **     7     0    30     0     0     0     7     7    31     7     0

   *LN2*     0     0     0     0     0     0     0     0     0     0     0
      **     0     0     0     0     0     0     0     0     0     0     0
   *LN3*     0     0     0     0     0     0     0     0     0     0     0
      **     0     0     0     0     0     0     0     0     0     0     0
   *LN4*     0     0     0     0     0     0     0     0     0     0     0
      **     0     0     0     0     0     0     0     0     0     0     0

   *STORED ROUTINE OBJECT*           0     0     0     0
   *DB ACCESS*                       0     0     0     0     0
                                     0     0     0     0
   *LN1*    26 2.67k     0 2.34k    88   331     0     0     0     0     0
      **     1    81     0    71          10     0     0     0     0     0

   *LN2*     0     0     0     0   278     0     0     0     0     0     0
      **                 0     0     8     0     0     0     0     0     0
   *LN3*     0     0     0     0     0     0     0     0      0     0     0
      **                 0     0     0     0     0     0      0     0     0
   *LN4*     0     0     0
      **     0     0     0

-----------------------------------------------------------------------------
FILE KIND   LOG KIND   FIRST                 LAST                         NUM
STJ         uap        2003/04/03 22:45:56   2003/04/03 22:56:24           33
-----------------------------------------------------------------------------

NO FILE KIND:LOG FILE NAME
  LOG KIND             FIRST                 LAST                         NUM
 1 STJ      :/tmp/stjdata/pdstj1
  sys                  2003/04/03 22:46:30   2003/04/03 22:46:30            2
  uap                  2003/04/03 22:45:56   2003/04/03 22:47:53           32
  sql                  2003/04/03 22:45:56   2003/04/03 22:47:53          898
  sop                    2003/04/03 22:45:56   2003/04/03 22:47:53          105
  dop                    2003/04/03 22:45:56   2003/04/03 22:47:52          113
  pcd                    2003/04/03 22:45:56   2003/04/03 22:47:53          759
  obj                    ****/**/** **:**:**   ****/**/** **:**:**            0
  sqh                    2003/04/03 22:45:56   2003/04/03 22:47:53          145
  buf                    2003/04/03 22:46:32   2003/04/03 22:46:32            3
  fil                    2003/04/03 22:46:32   2003/04/03 22:46:32           28
  dfw                    2003/04/03 22:45:58   2003/04/03 22:47:31           53
  idx                    2003/04/03 22:46:32   2003/04/03 22:46:32            1
  fsv                    ****/**/** **:**:**   ****/**/** **:**:**            0
  hba                    ****/**/** **:**:**   ****/**/** **:**:**            0

 2 STJ      :/tmp/stjdata/pdstj2
  sys                    2003/04/03 22:48:31   2003/04/03 22:56:35           10
  uap                    2003/04/03 22:47:54   2003/04/03 22:56:24           67
  sql                    2003/04/03 22:47:53   2003/04/03 22:56:24          623
  sop                    2003/04/03 22:47:53   2003/04/03 22:54:12           74
  dop                    2003/04/03 22:47:53   2003/04/03 22:56:23          102
  pcd                    2003/04/03 22:47:53   2003/04/03 22:56:24          516

  obj                    ****/**/** **:**:**   ****/**/** **:**:**            0
  sqh                    2003/04/03 22:47:53   2003/04/03 22:56:24          144
  buf                    2003/04/03 22:56:45   2003/04/03 22:56:45            3
  fil                    2003/04/03 22:56:45   2003/04/03 22:56:45           28
  dfw                    2003/04/03 22:48:04   2003/04/03 22:56:45           18
  idx                    2003/04/03 22:56:45   2003/04/03 22:56:45            1
  fsv                    ****/**/** **:**:**   ****/**/** **:**:**            0
  hba                    ****/**/** **:**:**   ****/**/** **:**:**            0
```

Explanation

1. Name of input statistics unload file or name of the directory containing the input statistics unload file (maximum of 58 bytes)
2. Output range (output start date/time to output end date/time)
3. Name of host requesting output of UAP or service statistical information
4. Edit period (collection start time to collection end time)
5. UAP name

   This is the name of the UAP for which statistical information was edited.
6. Service name

   This is the name of the service for which statistical information was edited. This name depends on the type of UAP that accessed HiRDB.[1]
7. UAP or transaction executions count

   This is the number of times the same UAP or transaction was executed according to the statistics log information.

   When UAP statistical information is obtained for UAP execution unit, the utility counts the number of times the UAP issued `DISCONNECT` as the executions count.

8. UAP or transaction normal terminations count

   This is the number of times the same UAP or transaction terminated normally according to the statistics log information.

9. UAP or transaction abnormal terminations count

   Number of times the last transaction executed by the UAP or transaction terminated in rolled-back status according to the statistics log information.

10. UAP or transaction execution time (milliseconds)

    This is the total execution time for the same UAP or transaction according to the statistics log information.

    If UAP statistical information is collected in units of UAP executions, the utility treats the time elapsed before the UAP issued `DISCONNECT` since it issued `CONNECT` as the execution time.

11. Average UAP or transaction execution time (milliseconds)

    This is the average execution time for the same UAP or transaction according to the statistics log information.

    If UAP statistical information is collected in units of UAP executions, the utility treats the time elapsed before the UAP issued `DISCONNECT` since it issued `CONNECT` as the execution time.

12. Size of table definition information buffer (bytes)

    This is the average size of the table definition information buffer used for the table definition information that was acquired by the same UAP or transaction according to the statistics log information.

13. Total lock release wait time for the UAP or transaction (milliseconds)[2]

    This is the total lock release wait time for the UAP or transaction.

14. Maximum lock release wait time for the UAP or transaction (milliseconds)[2]

    This is the maximum lock release wait time for the UAP or transaction.

15. Average lock release wait time for one UAP or transaction (milliseconds)[2]

    This is the average lock release wait time for one UAP or transaction.

16. Total CPU time for all servers within the transaction (milliseconds)
17. Average CPU time per UAP or transaction (`CTIME ÷ EXEC`) (milliseconds)
18. Total single server or front-end server processing time for the UAP or transaction (milliseconds)

19. Average single server or front-end server processing time per UAP or transaction (`SVEXT ÷ EXEC`) (milliseconds)

20. Number of SQL object acquisition requests

    This is the number of SQL object acquisition requests for the SQL statements issued by the same UAP or transaction within the specified time period.

21. Number of SQL object acquisition requests by UAP or transaction

    This is the average number of SQL object acquisition requests per UAP or transaction execution for the SQL statements that were issued by the same UAP or transaction within the specified time period.

22. SQL object buffer hits count

    This is the number of times a requested SQL object was found in the SQL object buffer for the SQL statements that were issued by the same UAP or transaction within the specified time period.

23. SQL object buffer hits count per UAP or transaction

    This is the average number of times per UAP or transaction execution a requested SQL object was found in the SQL object buffer for the SQL statements that were issued by the same UAP or transaction within the specified time period.

24. SQL object creations count

    This is the number of times an SQL object was created for the SQL statements issued by the same UAP or transaction within the specified time period.

25. SQL object creations count per UAP or transaction

    This is the average number of times per UAP or service execution that an SQL object was created for the SQL statements issued by the same UAP or transaction within the specified time period.

    If this value is large, the size of the SQL object buffer should be increased.

26. Maximum size of SQL objects created (bytes)

    This is the maximum size of the SQL objects created for the SQL statements that were issued by the same UAP or transaction within the specified time period.

27. `COMMIT` statement executions count

    This is the number of times the `COMMIT` statement issued by the same UAP or transaction was executed within the specified time period.

28. `ROLLBACK` statement executions count

    This is the number of times the `ROLLBACK` statement issued by the same UAP or transaction was executed within the specified time period.

29. Actually retrieved rows count

    This is the number of retrieved rows returned to the UAP or transaction by the `FETCH` and `SELECT` statements that were issued by the same UAP or transaction within the specified time period.

30. Actually deleted rows count

    This is the number of rows deleted by the `DELETE` statement that was issued by the same UAP or transaction within the specified time period.

31. Actually inserted rows count

    This is the number of rows inserted by the `INSERT` statement that was issued by the same UAP or transaction within the specified time period.

32. Actually updated rows count

    This is the number of rows updated by the `UPDATE` statement that was issued by the same UAP or transaction within the specified time period.

33. `SETUP` statement executions count

    This is the number of times preprocessing was executed by the same UAP or transaction within the specified time period.

34. `OPEN` statement executions count

    This is the number of times the `OPEN` statement was executed by the same UAP or transaction within the specified time period.

35. `FETCH` statement executions count

    This is the number of times the `FETCH` statement was executed by the same UAP or transaction within the specified time period.

36. `CLOSE` statement executions count

    This is the number of times the `CLOSE` statement was executed by the same UAP or transaction within the specified time period.

37. `DESCRIBE` statement executions count

    This is the number of times the `DESCRIBE` statement was executed by the same UAP or transaction within the specified time period.

38. `SELECT` statement executions count

    This is the number of times the `SELECT` statement was executed by the same UAP or transaction within the specified time period.

39. `INSERT` statement executions count

    This is the number of times the `INSERT` statement was executed by the same UAP or transaction within the specified time period.

40. UPDATE statement executions count

    This is the number of times the UPDATE statement was executed by the same UAP or transaction within the specified time period.

41. DELETE statement executions count

    This is the number of times the DELETE statement was executed by the same UAP or transaction within the specified time period.

42. LOCK statement executions count

    This is the number of times the LOCK statement was executed by the same UAP or transaction within the specified time period.

43. CREATE TABLE executions count

    This is the number of times CREATE TABLE was executed by the same UAP or transaction within the specified time period.

44. DROP TABLE executions count

    This is the number of times DROP TABLE was executed by the same UAP or transaction within the specified time period.

45. ALTER TABLE executions count

    This is the number of times ALTER TABLE was executed by the same UAP or transaction within the specified time period.

46. CREATE INDEX executions count

    This is the number of times CREATE INDEX was executed by the same UAP or transaction within the specified time period.

47. Number of times DROP INDEX issued by the UAP or transaction was executed

    This is the number of times DROP INDEX was executed by the same UAP or transaction within the specified time period.

48. COMMENT (TABLE) executions count

    This is the number of times COMMENT (TABLE) was executed by the same UAP or transaction within the specified time period.

49. COMMENT (COLUMN) executions count

    This is the number of times COMMENT (COLUMN) was executed by the same UAP or transaction within the specified time period.

50. CREATE SCHEMA executions count

    This is the number of times CREATE SCHEMA was executed by the same UAP or transaction within the specified time period.

51. `DROP SCHEMA` executions count

    This is the number of times `DROP SCHEMA` was executed by the same UAP or transaction within the specified time period.

52. `GRANT RDAREA` executions count

    This is the number of times `GRANT RDAREA` was executed by the same UAP or transaction within the specified time period.

53. `GRANT SCHEMA` executions count

    This is the number of times `GRANT SCHEMA` was executed by the same UAP or transaction within the specified time period.

54. `GRANT` access privilege executions count

    This is the number of times `GRANT` access privilege was executed by the same UAP or transaction within the specified time period.

55. `GRANT CONNECT` executions count

    This is the number of times `GRANT CONNECT` was executed by the same UAP or transaction within the specified time period.

56. `GRANT DBA` executions count

    This is the number of times `GRANT DBA` was executed by the same UAP or transaction within the specified time period.

57. `REVOKE RDAREA` executions count

    This is the number of times `REVOKE RDAREA` was executed by the same UAP or transaction within the specified time period.

58. `REVOKE SCHEMA` executions count

    This is the number of times `REVOKE SCHEMA` was executed by the same UAP or transaction within the specified time period.

59. `REVOKE` access privilege executions count

    This is the number of times `REVOKE` access privilege was executed by the same UAP or transaction within the specified time period.

60. `REVOKE CONNECT` executions count

    This is the number of times `REVOKE CONNECT` was executed by the same UAP or transaction within the specified time period.

61. `REVOKE DBA` executions count

    This is the number of times `REVOKE DBA` was executed by the same UAP or transaction within the specified time period.

62. `CREATE VIEW` executions count

    This is the number of times `CREATE VIEW` was executed by the same UAP or transaction within the specified time period.

63. `DROP VIEW` executions count

    This is the number of times `DROP VIEW` was executed by the same UAP or transaction within the specified time period.

64. `PURGE TABLE` statement executions count

    This is the number of times the `PURGE TABLE` statement was executed by the same UAP or transaction within the specified time period.

65. `CREATE PROCEDURE` executions count

    This is the number of times `CREATE PROCEDURE` was executed by the same UAP or transaction within the specified time period.

66. `DROP PROCEDURE` executions count

    This is the number of times `DROP PROCEDURE` was executed by the same UAP or transaction within the specified time period.

67. `ALTER PROCEDURE` executions count

    This is the number of times `ALTER PROCEDURE` was executed by the same UAP or transaction within the specified time period.

68. `CALL` statement executions count

    This is the number of times the `CALL` statement was executed by the same UAP or transaction within the specified time period.

69. `DESCRIBE` statement (`INPUT`) executions count

    This is the number of times the `DESCRIBE` statement (`INPUT`) was executed by the same UAP or transaction within the specified time period.

70. Other executions count

    This is the other executions count for the same UAP or transaction within the specified time period.

71. Number of stored procedure object acquisition requests

    This is the number of stored procedure object acquisition requests for the SQL statements issued by the same UAP or transaction within the specified time period.

72. Number of stored procedure object acquisition requests per UAP or transaction

    This is the average number of stored procedure object acquisition requests per UAP or transaction execution made for the SQL statements that were issued by the same UAP or transaction within the specified time period.

73. SQL object buffer hits count

    This is the number of times a requested stored procedure object was found in the SQL object buffer for the SQL statements that were issued by the same UAP or transaction within the specified time period.

74. SQL object buffer hits count per UAP or transaction

    This is the average number of times per UAP or transaction execution that a requested stored procedure object was found in the SQL object buffer for the SQL statements that were issued by the same UAP or transaction within the specified time period.

75. Maximum database input/output time (milliseconds)
76. Minimum database input/output time per operation (milliseconds)
77. Maximum number of work table files used by UAP
78. Maximum number of work table files added for use by UAP
79. Maximum size of work table file used by UAP (MB)
80. Estimated hash size required to expand all hash data at one time during hash join or subquery hash execution (KB)
81. Maximum packet size after level 1 packet division during hash join, subquery hash execution (KB)
82. Maximum packet size after level 2 packet division during hash join, subquery hash execution (KB)
83. Maximum packet size after level 3 packet division during hash join, subquery hash execution (KB)
84. Total input/output time for the database (milliseconds)
85. Number of times a data page, index page, or directory page was referenced[3]
86. Number of times a data page, index page, or directory page was updated[3]
87. Buffer hits count for data pages, index pages, or directory pages[3]
88. Buffer hit rate for data pages, index pages, or directory pages (`DIDHC ÷ DIDRC`) (%)
89. Actual number of `READ` operations on a data page, index page, or directory page
90. Actual number of `WRITE` operations on a data page, index page, or directory page[3]
91. Number of times a LOB column data page was referenced[3]

92. Number of times a LOB column data page was updated[3]
93. Buffer hits count for referencing LOB column data pages[3]
94. Buffer hits count for updating LOB column data pages
95. Buffer hit rate for referencing LOB column data pages (LBRHC ÷ LBRFC) (%)
96. Buffer hit rate for updating LOB column data pages (LBUHC ÷ LBUPC) (%)
97. Actual number of READ operations on LOB column data pages
98. Actual number of WRITE operations on LOB column data pages[3]
99. Global buffer flushes count[3]
100. READ waits count on global buffer[3]
101. WRITE waits count on global buffer[3]
102. Lock waits count on global buffer[3]
103. Number of times work table data was read from work table file to global buffer[3]
104. Number of times work table data was written from global buffer to work table file[3]
105. Number of times the contents of work table buffer were forcibly output to work table file due to shortage of work table buffer space[3]
106. With a table for which SEGMENT REUSE was specified, the number of times the mode changed from *new page allocate* to *free page reuse* and then changed back to *new page allocate* again because there was no reusable space[3]
107. With a table for which SEGMENT REUSE was specified, the number of times the mode changed from *new page allocate* to *free page reuse*[3]
108. Number of times data pages and index pages were referenced from the applicable UAP using the local buffer[3]
109. Number of times data pages and index pages were updated from the applicable UAP using the local buffer[3]
110. Local buffer hit count for data pages and index pages

     If a random access UAP's buffer hit count (LDIHC/LDIRC × 100) is low, you need to tune the applicable local buffer.[3]
111. Number of real READs on data pages and index pages from the applicable UAP using the local buffer[3]

If the prefetch facility is used, this value includes the number of pre-`READ`s achieved by prefetching. If the buffer hit rate (`LDIHC`/`LDIRC` × 100) is low, this value is greater than the number of real `READ`s.

112. Number of real `WRITE`s on data pages and index pages from the applicable UAP using the local buffer
113. Local buffer flush count (number of times the buffer contents were discarded to read new pages)[3]
114. Number of batch pre-read requests issued to asynchronous `READ` processes when the asynchronous `READ` facility is used[3]
115. Number of times synchronization waits occurred during batch pre-read operation executed by asynchronous `READ` processes when the asynchronous `READ` facility is used[3]
116. Synchronous wait time during batch pre-read operation executed by asynchronous `READ` processes when the asynchronous `READ` facility is used (milliseconds)[3]
117. Synchronous `READ` time during the first top-page batch read operation when the asynchronous `READ` facility is used (milliseconds)[3]
118. Maximum comparison count when a hash row partitioning table was searched during hash join, subquery hash execution
119. Average comparison count when a hash row partitioning table was searched during hash join, subquery hash execution[3]
120. Total value of each item per UAP or transaction execution
121. Average value of each item per UAP or transaction execution
122. Grand total for each item

[1] The following table shows the service name that is displayed for each type of UAP:

UAP type		Displayed service name
Open/TP1 UAP	Service requested from Open/TP1 SUP (service using program) to SPP (service provider program)	Name of the corresponding service
	Service requested from TP1/ Message Control to MHP (message handling program)	
	Other	31 consecutive asterisks (`*`)
Other types of UAP		

2 For a HiRDB/Parallel Server, this is the total lock release wait time in the transaction (including parallel processing).

3 If the information is output for each transaction, this is the cumulative `CONNECT` value.

14.3.4 SQL statistical information

The following shows the statistical information about SQL:

```
pdstedit VV-RR          ***** SQL INFORMATION *****
INPUT          :/tmp/pdstj1 [1]
OUTPUT RANGE   :**/**/** **:**:** - **/**/** **:**:** [2]
-----------------------------------------------------------------------------
HOST = test [3]
-----------------------------------------------------------------------------
FES = fes01 [4]
-----------------------------------------------------------------------------

EDIT TIME 2001/09/30 00:00:00 - 2001/09/31 00:00:00 [5]
UAP NAME [6]
  SERVICE NAME [7]                 [8]   [9]   [10]  [11]
                                   PDATA PDAVG BES#  B#AVG
------------------------------     ----- ----- ----- -----
         [12] [13]     [14]  [15]  [16]  [17]  [18]  [19]  [20]  [21]
         KIND SERV     EXEC  TTIME FTCR  INSR  UPDR  DELR  CRTL  DRPL
         ---- -------- ----- ----- ----- ----- ----- ----- ----- -----


uap01
  service01
                                    745k   270     0     0
   *CNT* SQL           2.82k  272M     0   532     0     0     0     0 [22]
      **                     96.5k     0     0     0     0     0     0 [23]
   *CNT* FSQL              0     0     0     0     0     0 [22]
      **                         0     0     0     0     0 [23]
uap02
  service02
                                    467k    60     0     0
   *CNT* SQL           7.90k 79.0M 7.55k     0     0     0     0     0
      **                     10.0k     1     0     0     0     0     0
   *CNT* FSQL            750 5.60M 7.55k
      **                     12.0k     2
```

```
uap03
  service03
                                  11.2M   461     0     0
   *CNT* SQL          25.5k  310M 5.39k     0     0     0     0     0
      **                    12.2k     0     0     0     0     0     0
   *CNT* FSQL         2.32k 27.9M 5.39k
      **                    12.0k     2
uap04
  service04
                                  7.50M   534     0     0
   *CNT* SQL          14.7k  192M 5.99k   716   728   716     0     0
      **                    13.1k     0     0     0     0     0     0
   *CNT* FSQL           445 5.82M 1.99k
      **                    13.0k     4

*TOTAL*[24]                       19.9M   410     0     0
   *CNT* SQL          51.0k  854M 18.9k 1.25k   728   716     0     0
      **                    16.8k     0     0     0     0     0     0
   *CNT* FSQL         3.52k 39.3M 14.9k
      **                    11.2k     4
-----------------------------------------------------------------------------
FILE KIND   LOG KIND   FIRST                 LAST                        NUM
STJ         sql        2001/09/30 10:36:37   2001/09/30 12:34:03       50971
-----------------------------------------------------------------------------
```

Explanation

1. Name of input statistics unload file or name of the directory containing the input statistics unload file (maximum of 58 bytes)
2. Output range (output start date/time to output end date/time)
3. Name of host requesting output of SQL statistical information
4. Name of front-end server that executed SQL
5. Edit period (collection start time to collection end time)
6. UAP name

   This is the name of the UAP for which statistical information was edited.
7. Service name

   This is the name of the service for which statistical information was edited. This name depends on the type of UAP that accessed HiRDB.*
8. Total size of SQL objects created (bytes)

   This is the total size of SQL objects created by the SQL statements that were issued within the UAP or service.
9. Average size of SQL objects created (bytes)

   This is the average size of SQL objects created for the SQL statements that were issued within the UAP or service.

This value is obtained by the formula size of SQL objects created ÷ SQL executions count.

10. Number of back-end servers used

    This is the number of back-end servers used for the SQL statements that were issued within the UAP or service.

11. Average number of back-end servers used

    This is the average number of back-end servers used for the SQL statements that were issued within the UAP or service.

12. Type of output information

    In the case of SQL, this is information about SQL. In the case of FSQL, this information is about the foreign server.

13. Information used by the system (not by users)

14. SQL executions count

    If the type of output information is SQL, this is the number of times SQL statements issued within the UAP or service were executed. If the type of output information is FSQL, this is the number of times processing requests were issued to the foreign servers.

15. SQL execution time (microseconds)

    If the type of output information is SQL, this is the total processing time for SQL statements issued within the UAP or service. If the type of output information is FSQL, this is the total processing time for the foreign servers. Note that this is the database access adaptor's processing time, not the foreign server's processing time.

16. Number of rows retrieved

    If the type of output information is SQL, this is the number of rows retrieved by the `FETCH` and `SELECT` statements issued within the UAP or service.

17. Number of rows inserted

    This is the number of rows inserted by the `INSERT` statement issued within the UAP or service.

18. Number of rows updated

    This is the number of rows updated by the `UPDATE` statement issued within the UAP or service.

19. Number of rows deleted

This is the number of rows deleted by the `DELETE` statement issued within the UAP or service.

20. Number of times work tables were created

    This is the number of times work tables were created internally by the SQL statements issued within the UAP or service.

21. Number of times work tables were deleted

    This is the number of times work tables created internally by the SQL statements issued within the UAP or service were deleted. If an error has occurred, the correct number of times work tables were deleted may not be displayed.

22. Total of each item

    If the type of output information is FSQL and the HiRDB External Data Access facility is not used, `0` or `****` is displayed.

23. If the type of output information is SQL, this is the average value (total of each item/number of executions). If the type of output information is FSQL, this is the average value (total of each item/number of processing requests to the foreign server). If the HiRDB External Data Access facility is not used, `0` or `****` is displayed.

24. Grand totals for each item

* The following table shows the service name that is displayed for each type of UAP:

UAP type		**Displayed service name**
Open/TP1 UAP	Service requested from Open/TP1 SUP (service using program) to SPP (service provider program)	Name of the corresponding service
	Service requested from TP1/Message Control to MHP (message handling program)	
	Other	31 consecutive asterisks (`*`)
Other types of UAP		

14.3.5 Global buffer pool statistical information

The following shows the statistical information about the global buffer pool:

```
pdstedit VV-RR(Object Option) ***** GLOBAL BUFFER INFORMATION *****
INPUT          :/tmp/stjdata [1]
OUTPUT RANGE   :**/**/** **:**:** - **/**/** **:**:** [2]
-----------------------------------------------------------------------------
HOST = test [3]
-----------------------------------------------------------------------------

EDIT TIME 2003/04/03 22:00:00 - 2003/04/03 23:00:00 [4]
               [5]   [6]   [7]   [8]  [9]    [10]  [11]  [12] [13]  [14]
         *LN1* SYNCW MAXB  UPGET UPHIT(HIT)  UPFLS RFGET RFHIT(HIT) RFFLS
               ----- ----- ----- ----- ---   ----- ----- ----- ---  -----
               [15]  [16]  [17]  [18]  [19]  [20]  [21]  [22] [23]  [24]
         *LN2* READ  WRITE WAITR WAITW WAITL BFINS PRRED PRHIT(HIT) PRINS
               ----- ----- ----- ----- ----- ----- ----- ----- ---  -----

               [25]  [26]  [27]  [28]  [29]  [30]  [31]  [32]       [33]
         *LN3* GBHIT CURRF CURUP TRGUP SYNCC PRRDR LRDRC LWTRC      LBBKR
               ----- ----- ----- ----- ----- ----- ----- -----      -----
               [34]  [35]  [36]  [37]  [38]     [39]  [40]     [41]  [42]
         *LN4* LBBKW CINSM CFMAX CFAVG SLEPC    SLEPR SLEPA    SPINR SPINA
               ----- ----- ----- ----- -------- ----- -------- ----- -----
               [43]  [44]  [45]  [46]
         *LN5* SYNCL SYNCB ALTRW ALTUW
               ----- ----- ----- -----

SERVER :  sds [47]
  *BUFFER NAME:bp01 [48]          BUFFER:   10 [49]
         *LN1*     0     2   160   114( 52)     14    60    60( 27)     8
         *LN2*     6    72     0     2     0     0     0     0(  0)     0
         *LN3*    79     0     0     0     1     0     0     0          0
         *LN4*     0     0     0  ****  0.0e+00   0.0  *******   0.1 2.05k
         *LN5*     0     0     0     0

  *BUFFER NAME:bp02               BUFFER:   10
         *LN1*     0     3 1.51k   933(  4)    278 20.8k 19.7k( 88)   657
         *LN2* 1.30k   607     0    36     0     0     0     0(  0)     0
         *LN3*    93     0     0     0     1     0     0     0          0
         *LN4*     0     0     0  ****  0.0e+00   0.0  *******   0.1 2.05k
         *LN5*     0     0     0     0

  *BUFFER NAME:bp03               BUFFER:   10
         *LN1*     0     4 1.22k   647( 15)     50 3.15k 1.50k( 34) 1.73k
         *LN2* 2.22k   561     0    69     1     0     0     0(  0)     0
         *LN3*    49     0     0     0     1     0     0     0          0
         *LN4*     0     0     0  ****  0.0e+00   0.0  *******   0.4 2.05k
         *LN5*     0     0     0     0
  *TOTAL* [50]
         *LN1*     0     4 2.89k 1.69k(  6)    342 24.0k 21.3k( 79) 2.39k
         *LN2* 3.52k 1.24k     0   107     1     0     0     0(  0)     0
         *LN3*    85     0     0     0     3     0     0     0          0
         *LN4*     0     0     0  ****  0.0e+00   0.0  *******   0.2 2.05k
         *LN5*     0           0     0
------------------------------------------------------------------------------
FILE KIND   LOG KIND   FIRST                 LAST                          NUM
STJ         buf        2003/04/03 22:46:32   2003/04/03 22:56:45             6
------------------------------------------------------------------------------



NO FILE KIND:LOG FILE NAME
  LOG KIND             FIRST                 LAST                          NUM
 1 STJ      :/tmp/stjdata/pdstj01
  sys                  2003/04/03 22:46:30   2003/04/03 22:46:30             2
  uap                  2003/04/03 22:45:56   2003/04/03 22:47:53            32
  sql                  2003/04/03 22:45:56   2003/04/03 22:47:53           898
  sop                  2003/04/03 22:45:56   2003/04/03 22:47:53           105
  dop                  2003/04/03 22:45:56   2003/04/03 22:47:52           113
  pcd                  2003/04/03 22:45:56   2003/04/03 22:47:53           759

  obj                  ****/**/** **:**:**   ****/**/** **:**:**             0
  sqh                  2003/04/03 22:45:56   2003/04/03 22:47:53           145
  buf                  2003/04/03 22:46:32   2003/04/03 22:46:32             3
  fil                  2003/04/03 22:46:32   2003/04/03 22:46:32            28
  dfw                  2003/04/03 22:45:58   2003/04/03 22:47:31            53
  idx                  2003/04/03 22:46:32   2003/04/03 22:46:32             1
  fsv                  ****/**/** **:**:**   ****/**/** **:**:**             0
  hba                  ****/**/** **:**:**   ****/**/** **:**:**             0

 2 STJ      :/tmp/stjdata/pdstj02
  sys                  2003/04/03 22:48:31   2003/04/03 22:56:35            10
  uap                  2003/04/03 22:47:54   2003/04/03 22:56:24            67
  sql                  2003/04/03 22:47:53   2003/04/03 22:56:24           623
  sop                  2003/04/03 22:47:53   2003/04/03 22:54:12            74
  dop                  2003/04/03 22:47:53   2003/04/03 22:56:23           102
  pcd                  2003/04/03 22:47:53   2003/04/03 22:56:24           516

  obj                  ****/**/** **:**:**   ****/**/** **:**:**             0
  sqh                  2003/04/03 22:47:53   2003/04/03 22:56:24           144
  buf                  2003/04/03 22:56:45   2003/04/03 22:56:45             3
  fil                  2003/04/03 22:56:45   2003/04/03 22:56:45            28
  dfw                  2003/04/03 22:48:04   2003/04/03 22:56:45            18
  idx                  2003/04/03 22:56:45   2003/04/03 22:56:45             1
  fsv                  ****/**/** **:**:**   ****/**/** **:**:**             0
  hba                  ****/**/** **:**:**   ****/**/** **:**:**             0
```

Explanation

1. Name of input statistics unload file or name of the directory containing the input statistics unload file (maximum of 58 bytes)
2. Output range (output start date/time to output end date/time)
3. Name of host requesting output of global buffer pool statistical information

4. Edit period (collection start time to collection end time)

5. Number of pages output during synchronization point dump

   This is the maximum number of pages output from the global buffer pool to the HiRDB file during a synchronization point dump.

   If this value is large, many pages may be updated in the corresponding global buffer pool, and the updated page output rate may be low during a deferred write trigger, resulting in many pages kept in the global buffer pool. Evaluate whether the updated page output rate can be reduced during a deferred write trigger.

6. Maximum number of concurrently requested buffer sectors

   This is the number of global buffer pool acquisition requests issued concurrently from multiple transactions.

   If this value is almost the same as the number of global buffer sectors and the buffer shortages count is not 0, the buffer may be insufficient.

7. Update `GET`s count

   This is the number of times buffer acquisition was requested to the global buffer pool for updating purposes.

8. Update buffer hits count

   This is the number of times a page requested for the global buffer pool for updating purposes was found as the updated buffer.

9. Update buffer hit rate (%)

   This is the ratio of update buffer hits count to the total of update `GET`s count and reference `GET`s count. It is the probability of finding a requested page (for updating or referencing purposes) as updated buffer in the global buffer pool without executing a physical input/output operation.

   This value is obtained from the following formula:
   ```
   Update buffer hit rate = (update buffer hits count ÷
   (reference GETs count + update GETs count)) × 100
   ```

   If this value is small, the deferred write processing may not be effective. In this case, you need to increase the number of global buffer pool sectors or reduce the updated page output rate during a deferred write trigger.

10. Update buffer flushes count

    This is the number of times the buffer was invalidated after writing its contents to a HiRDB file to create an empty buffer for reading new pages, because the referenced buffer had been updated.

    If this value is large, the deferred write processing may not be effective.

11. Reference `GET`s count

    This is the number of times buffer acquisition was requested for the global buffer pool for referencing purposes.

12. Reference buffer hits count

    This is the number of times a page requested for the global buffer pool for referencing purposes was found as the referenced (not updated) buffer.

13. Reference buffer hit rate (%)

    This is the ratio of reference buffer hits count to the total of reference `GET`s count and update `GET`s count. It is the probability of finding a requested page (for referencing or updating purposes) as referenced buffer in the global buffer pool without executing a physical input/output operation.

    This value is obtained from the following formula:
    ```
    Reference buffer hit rate = (reference buffer hits count ÷
    (reference GETs count + update GETs count)) × 100
    ```

    The reference buffer hit rate should be about 80%. If it is less than 80%, you need to increase the number of buffer sectors.

14. Reference buffer flushes count

    This is the number of times the buffer was invalidated to create an empty buffer for reading new pages, because the referenced buffer had not been used since that time.

15. Actual `READ`s count

    This is the number of times pages were read from HiRDB file.

16. Actual `WRITE`s count

    This is the number of times pages were written to a HiRDB file.

17. Input waits count

    This is the number of times wait status occurred because a requested page was being read from a HiRDB file by another user.

18. Output waits count

    This is the number of times wait status occurred because a requested page was being written to a HiRDB file by another user.

19. Buffer lock release waits count

    This is the number of times wait status occurred because a requested page was being used by another user.

20. Buffer shortages count

This is the number of times there was no available buffer because all buffers (defined buffer sectors) were being used for referencing or update processing.

This value must be 0.

21. Number of prefetched input pages

    This is the number of pages read in memory as being subject to prefetch processing.

22. Prefetch hits count

    This is the hits count on the pages that were read in memory as being subject to prefetch processing.

23. Prefetch hit rate (%)

    This is the ratio of the prefetch hits count to the number of prefetched input pages.

24. Prefetch resource shortages count

    This is the number of times a shortage occurred on the resources required for prefetch processing.

25. Global buffer hit rate (%)
26. Number of current reference buffers
27. Number of current update buffers
28. Number of update buffers that triggers output operation when deferred write trigger is on
29. Number of synchronization point dumps that occurred
30. Number of prefetch `READ` requests
31. Number of LOB global buffer `READ` requests
32. Number of LOB global buffer `WRITE` requests
33. Number of LOB global buffer pages input in batch mode
34. Number of LOB global buffer pages output in batch mode
35. Information used by the system (not for the user)
36. Information used by the system (not for the user)
37. Information used by the system (not for the user)
38. Information used by the system (not for the user)
39. Percentage of buffer contention-lock release waits resulting from buffer lock processing (%)
40. Information used by the system (not for the user)

41. Information used by the system (not for the user)
42. Information used by the system (not for the user)
43. Buffer pool lock exclusive time during synchronization point processing (microseconds)
44. Number of buffers processed within buffer pool lock exclusive time during synchronization point processing
45. Take-over count of database write processing by reference request hit during synchronization point processing
46. Take-over count of database write processing by update request hit during synchronization point processing
47. Server name

    This is the server requesting output of global buffer pool statistical information.
48. Global buffer pool name

    This is the name of the global buffer pool for which statistical information was edited.
49. Number of buffer sectors in the global buffer pool

    This is the number of buffer sectors in the corresponding global buffer pool.
50. Totals for each item

14.3.6 Statistical information about HiRDB files for database manipulation

The following shows the statistical information about HiRDB files for database manipulation:

```
pdstedit VV-RR           ***** HiRDB FILE INFORMATION *****
INPUT          :/tmp/pdstj1 [1]
OUTPUT RANGE   :**/**/** **:**:** - **/**/** **:**:** [2]
-----------------------------------------------------------------------------
HOST = test [3]
-----------------------------------------------------------------------------

EDIT TIME 1996/10/30 00:00:00 - 1996/10/31 00:00:00 [4]
               [5]    [6]    [7]    [8]    [9]    [10]   [11]   [12]
         *LN1* SYNC-R SYNC-W AIO-R  AIO-W  LISTIO OPEN   CLOSE  IOERR
               -----  -----  -----  -----  -----  -----  -----  -----
```

```
SERVER : sds [13]
 *FILE NAME : /DB6G/si1/kado/single/pddir/iospt0/h001
   *RDAREA NAME :RDMASTER [15]
         *LN1*    67    142      0     85     85      8      7      0
 *FILE NAME : /DB6G/si1/kado/single/pddir/iospt0/h002
   *RDAREA NAME :RDDIR01
         *LN1*    67    116      0     81     81      8      7      0

 *FILE NAME : /DB6G/si1/kado/single/pddir/iospt0/h003
   *RDAREA NAME :RDDIC01
         *LN1*   824  2.15k      0  2.00k  2.00k     19     19      0
 *FILE NAME : /DB6G/si1/kado/single/pddir/iospt1/u001
   *RDAREA NAME :RU01
         *LN1*   256    483      0    401    401     23     22      0

 *FILE NAME : /DB6G/si1/kado/single/pddir/iospt2/u002
   *RDAREA NAME :RU02
         *LN1*   261    501      0    416    416     26     25      0
 *FILE NAME : /DB6G/si1/kado/single/pddir/iospt3/u003
   *RDAREA NAME :RU03
         *LN1*   254    467      0    384    384     23     22      0

 *FILE NAME : /DB6G/si1/kado/single/pddir/iospt4/u001
   *RDAREA NAME :RULOB01
         *LN1*    16     43      0     33     33      4      4      0
 *FILE NAME : /DB6G/si1/kado/single/pddir/iospt4/u002
   *RDAREA NAME :RULOB02
         *LN1*    16     43      0     33     33      4      4      0

 *FILE NAME : /DB6G/si1/kado/single/pddir/iospt4/u011
   *RDAREA NAME :RUBLOB01
         *LN1*    30     59      0      0      0      7      7      0
 *FILE NAME : /DB6G/si1/kado/single/pddir/iospt4/u012
   *RDAREA NAME :RUBLOB02
         *LN1*     0      0      0      0      0      0      0      0

 *TOTAL* [16]
         *LN1* 1.79k  4.00k      0  3.43k  3.43k    122    117      0
-----------------------------------------------------------------------------
FILE KIND   LOG KIND   FIRST                 LAST                        NUM
STJ         fil        1996/10/30 10:38:19   1996/10/30 12:31:45         440
-----------------------------------------------------------------------------
```

Explanation

1. Name of input statistics unload file or name of the directory containing the input statistics unload file (maximum of 58 bytes)
2. Output range (output start date/time to output end date/time)
3. Name of host requesting output of statistical information about HiRDB files for database manipulation
4. Edit period (collection start time to collection end time)
5. Synchronous `READ`s count

This is the number of times pages were read from the HiRDB file in the synchronous mode.

6. Synchronous `WRITE`s count

   This is the number of times pages were written to the HiRDB file in the synchronous mode.

7. Information used by the system (not by users).

8. Asynchronous `WRITE`s count

   This is the number of times pages were written to the HiRDB file in the asynchronous mode.

   This value is the sum of the following values:

   - Number of times pages were written to the HiRDB file during deferred write trigger processing
   - Number of times pages were written to the HiRDB file during synchronization point dump

9. Information used by the system (not by users).

10. Open operations count

    This is the number of times HiRDB file open processing was executed.

11. Close operations count

    This is the number of times HiRDB file close processing was executed.

12. Input/output errors count

    This is the number of times an error occurred during HiRDB file input/output operations.

13. Server name

    This is the name of server requesting output of statistical information about HiRDB files for database manipulation.

14. Name of HiRDB file

    This is the name of the HiRDB for which statistical information was edited.

15. Name of RDAREA

    This is the name of RDAREA using the corresponding HiRDB file.

16. Totals for each item

14.3.7 Deferred write processing statistical information

The following shows the statistical information about deferred write processing:

```
pdstedit VV-RR(Object Option) ***** DEFERRED WRITE INFORMATION *****
INPUT          :/tmp/pdstj1 [1]
OUTPUT RANGE   :**/**/** **:**:** - **/**/** **:**:** [2]
-----------------------------------------------------------------------------
HOST = test [3]
-----------------------------------------------------------------------------
EDIT TIME 2004/03/11 10:00:00 - 2004/03/11 12:00:00 [4]
 [5]       [6]         [7]   [8]   [9]   [10]  [11]        [12]  [13]
 SERVER    EXEC        TRG   PRSYC SYNC  DBSYC RDSYC       END   CANCL
           -----       ----- ----- ----- ----- -----       ----- -----
                       [14]  [15]  [16]        [17]  [18]
                       PMAX  PMIN  PAVG        TOTAL AVG
                       ----- ----- -----       ----- -----
 sds         103          15    44    44     0     0          32    12
                           1     1     1       3.56k    35
-----------------------------------------------------------------------------
FILE KIND   LOG KIND   FIRST                LAST                         NUM
STJ         dfw        2004/03/11 10:15:08   2004/03/11 12:33:31          103
-----------------------------------------------------------------------------
```

Explanation

1. Name of input statistics unload file or name of the directory containing the input statistics unload file (maximum of 58 bytes)
2. Output range (output start date/time to output end date/time)
3. Name of host requesting output of deferred write processing statistical information
4. Edit period (collection start time to collection end time)
5. Name of server requesting output of deferred write processing statistical information
6. Number of deferred write requests

   This is the number of times deferred write processing (delayed write processing) was executed within the specified time period.
7. Trigger count

   This is the number of times within the specified time period that deferred write processing was requested when the percentage of updated pages in the global buffer (ratio of updated buffer sectors to the total number of buffer sectors) reached a specified value.
8. Pre-synchronization count

   This is the number of times within the specified time period deferred write processing was requested because a specified value was reached before the synchronization point dump interval (the number of times data that was output to the system log file reached a specified value before the next synchronization point dump).

9. Synchronization points count

   This is the number of database synchronization points that occurred (the number of times a synchronization point was set for database integrity assurance) within the specified time period.

10. Database synchronization points count

    This is the number of processes for applying update buffers to the database at a synchronization point where synchronization point processing cannot be validated.

11. RDAREA synchronization points count

    This is the number of processes for outputting to disk all update pages for RDAREAs.

12. Pre-synchronization completion status (outputs count)

    This is the number of times within the specified time period all the update buffer was output to the HiRDB file before the next synchronization point dump when deferred write processing was requested because a specified value was reached before the synchronization point dump interval (the number of times data that was output to the system log file reached a specified value before the next synchronization point dump).

13. Pre-synchronization completion status (cancellations count)

    When deferred write processing was requested because a specified value was reached before the synchronization point dump interval (the number of times data that was output to the system log file reached a specified value before the next synchronization point dump), this is the number of times within the specified time period that deferred write processing was cancelled because the utility was unable to write all the update buffer to the HiRDB file before the next synchronization point dump.

14. Concurrency level for each disk volume (maximum value)

    This is the maximum concurrency level of input/output processing executed for each disk for deferred write processing within the specified time period.

    The utility executes input/output processing concurrently if all the following conditions are satisfied:

    - The HiRDB file is a character special file.
    - Multiple disks are used.

    If there are only one maximum value and one minimum value, input/output operations may be concentrated on a specific disk (HiRDB file). You should use multiple disks to create RDAREAs subject to update processing.

15. Concurrency level for each disk volume (minimum value)

    This is the minimum concurrency level of input/output processing executed for each disk for deferred write processing within the specified time period.

    If there are only one maximum value and one minimum value, input/output operations may be concentrated on a specific disk (HiRDB file). You should use multiple disks to store one or more RDAREAs subject to update processing.

16. Concurrency level for each disk volume (average value)

    This is the average concurrency level of input/output processing executed for each disk for deferred write processing within the specified time period.

17. Number of output pages (total)

    This is the total number of pages output to the HiRDB file during deferred write processing (delayed write processing) within the specified time period.

18. Number of output pages (average)

    This is the average number of pages output to the HiRDB file during deferred write processing (delayed write processing) within the specified time period.

14.3.8 Index statistical information

The following shows the statistical information about the index:

```
pdstedit VV-RR          ***** INDEX INFORMATION *****
INPUT          :/tmp/pdstj1 [1]
OUTPUT RANGE   :**/**/** **:**:** - **/**/** **:**:** [2]
CONTROL FILE     :CONTROL1 [18]
                                      [19]
FILE GROUP     :SERV1  : FILE1 FILE2 FILE3 FILE4 FILE5
                         FILE6 FILE7 FILE8 FILE9 FILE10
-----------------------------------------------------------------------------

EDIT TIME 1996/10/30 00:00:00 - 1996/10/31 00:00:00 [3]
             [4]   [5]   [6]   [7]   [8]   [9]   [10]  [11]
    *LN1*    C_SLK W_SLK XLOCK DEADL UNQCK UNQER REPOS REPG
             ----- ----- ----- ----- ----- ----- ----- -----
             [12]  [13] [14]
    *LN2*    SP_NM LVL ( NUM )
             ----- ---  -----
```

```
SERVER : sds [15]
 *INDEX_ID :      196611 [16]
  *RDID    :           3 [17]
    *LN1*    4.32k     0     0     0     0     0     0     0
    *LN2*        0
 *INDEX_ID :      196612
  *RDID    :           3
    *LN1*    5.31k     0     0     0     0     0     0     0
    *LN2*        0

 *INDEX_ID :      196613
  *RDID    :           3
    *LN1*    4.64k     0 1.35k     0   684     0     0     0
    *LN2*        0
 *INDEX_ID :      196614
  *RDID    :           3
    *LN1*       12     0 2.03k     0   684     0     0     0
    *LN2*        0

 *INDEX_ID :      196615
  *RDID    :           3
    *LN1*    4.86k     0 1.94k     0     0     0     0     0
    *LN2*        0
------------------------------------------------------------------------------
FILE KIND   LOG KIND   FIRST                 LAST                        NUM
STJ         idx        1996/10/30 10:38:19   1996/10/30 12:31:45          44
FJ          idx        ****/**/** **:**:**   ****/**/** **:**:**           0
------------------------------------------------------------------------------



NO FILE KIND:LOG FILE NAME
  LOG KIND             FIRST                 LAST                        NUM
 1 STJ      :/tmp/pdstj1
  sys                  1996/10/30 10:37:33   1996/10/30 12:33:07         112
  uap                  1996/10/30 10:36:38   1996/10/30 12:34:04        1854
  sql                  1996/10/30 10:36:37   1996/10/30 12:34:03       50971
  sop                  ****/**/** **:**:**   ****/**/** **:**:**           0
  dop                  ****/**/** **:**:**   ****/**/** **:**:**           0
  pcd                  ****/**/** **:**:**   ****/**/** **:**:**           0
  buf                  1996/10/30 10:38:19   1996/10/30 12:31:45          44
  fil                  1996/10/30 10:38:19   1996/10/30 12:31:45         440
  dfw                  1996/10/30 10:38:12   1996/10/30 12:33:31         103
  idx                  1996/10/30 10:38:19   1996/10/30 12:31:45          44
```

Explanation

1. Name of input statistics unload file or name of the directory containing the input statistics unload file (maximum of 58 bytes)
2. Output range (output start date/time to output end date/time)
3. Edit period (collection start time to collection end time)
4. Key value reference locks count in `CHECK` mode

   This is the number of times reference lock (PR) was acquired for a key value in the CHECK mode (if another user has a lock, this mode does not wait until the lock is released).

5. Key value reference locks count in WAIT mode

   This is the number of times reference lock (PR) was acquired for a key value in the WAIT mode (if another user has a lock, this mode waits until the lock is released).

6. Key value update locks count

   This is the number of times an update lock (EX) was acquired for a key value.

7. Key value update deadlocks count

   This is the number of times deadlock resulted from acquisition of an update lock (EX) for a key value.

8. Unique checks count

   This is the number of times unique value checking was performed by the INSERT or UPDATE statement. This information is obtained only for a unique index.

9. Unique errors count

   This is the number of times unique value checking performed by the INSERT or UPDATE statement resulted in an error because a key with the same value was found. This information is obtained only for a unique index.

   If this value is large, you need to reevaluate whether the unique index is required.

10. Repositioning count (current position)

    This is the number of times repositioning (correction) was performed because the position of the referenced key value was changed by an addition or deletion made to another key value in the same page by another user.

11. Repositioning count (current page)

    This is the number of times repositioning (correction) was performed, because the page containing the searched key value changed due to a page split caused by another user.

12. Splits count (index split processing)[2]

    This is the number of times split (index split processing)[1] occurred.

    If this value is large, you need to increase the percentage of PCTFREE for the index. If you are adding a large amount of data, you should use the database load utility (pdload).

13. Number of levels affected by split[2]

14. Splits count for each level affected by split processing

    This is the sum of the splits counts of the upper-level pages for each level affected by split processing.

    If the splits count is large at a higher level, the key value may be too large for the size of the index page or the percentage of `PCTFREE` may be too large for the index.

15. Server name

    This is the name of the server requesting output of index statistical information.

16. Index ID

    This is the number of the index for which statistical information was edited.

    You can obtain the index name by searching the data dictionary table (`SQL_INDEXES` table) on the basis of the index number displayed.

17. RDAREA ID

    This is the number of the RDAREA storing the index for which statistical information was edited.

    You can obtain the RDAREA name by searching the data dictionary table (`SQL_RDAREAS` table) on the basis of the RDAREA number displayed.

18. Name of the control statement file (up to 58 bytes)

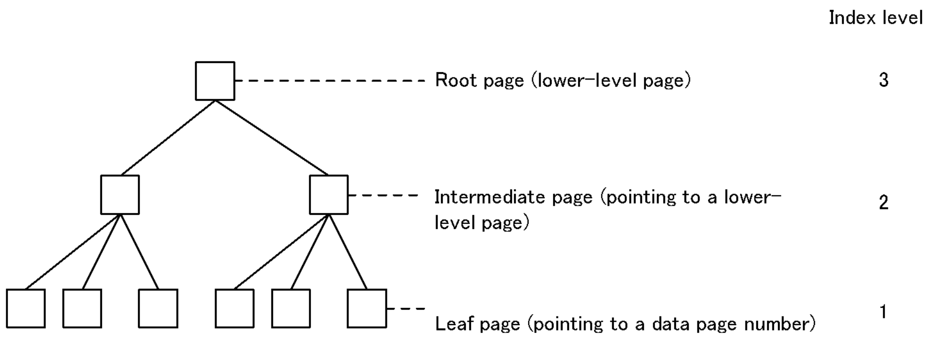
    This is the name of the control statement file specified by the `-d` option.

19. Server name and file group name

    These are the server name in the control statement file specified by the `-d` option (name of server to be analyzed[3]) and the corresponding file group names.

Notes

1. 13 and 14 are displayed as many times as there are split levels.
2. 12 through 14 are displayed only for an unload log file or if a system log file was read with the `-d` option specified.
3. The following explains the levels affected by split processing:


Index level
Root page (lower-level page)
3
Intermediate page (pointing to a lower-level page)
2
Leaf page (pointing to a data page number)
1

Explanation

The first split occurs on a leaf page. As a result of this split, a new page is created. If this split processing is completed after the new page is registered in the upper level page (intermediate page), this split affects level 2 (`LVL`), so the number of levels affected by the split is set to 2 (`NUM`).

If the split affects the page above index level 2, another new page is registered in the upper level page. If there is no space in this page, another split occurs on the upper level page. In this case, the number of levels affected by the split (`LVL`) becomes 3, and the count (`NUM`) is set to 3.

[1] A split is processing that divides the contents of a page and then adds the latter half of the contents to an unused page; this occurs when there is not enough space to add a key value in a used page in the index storage RDAREA.

[2] This information is displayed only when system log is read.

[3] The following servers are subject to analysis:

- The server with a name specified both in the `-s` option and in the control statement file
- All servers specified in the control statement file that are not specified in the `-s` option

14.4 Output of statistical information to a DAT-format file

Statistical information output to a DAT-format file can be used as the input file to other table operation software.

14.4.1 Data storage format of a DAT-format file

Statistical information output to a DAT-format file is edited and stored as shown in Table 14-3.

Table 14-3: Data editing format for DAT-format file

Data format	Storage method
Missing value	Represented as two consecutive double quotation marks ("").
Character data	Enclosed in double quotation marks (").
Numeric data	Stored as is.
Overflow data	4 consecutive asterisks (*)*.
Uneditable data	10 consecutive asterisks (*)*

* When `erl` is specified in the `-e` option, one asterisk (`*`) is stored.

14.4.2 DAT-format file unit

When statistical information is output to a DAT-format file, one file is created for each type of statistical information. The file is created under a user-specified directory. Table 14-4 lists the names of DAT-format files.

Table 14-4: Names of DAT-format files

Edited information	Output unit	File name
System activity statistical information	Activity information in the log for one system	`sys_DAT`
UAP statistical information	Execution of one UAP or one service	`uap_DAT`
SQL statistical information	Execution of one SQL	`sql_DAT`
Global buffer pool statistical information	Each synchronization point	`buf_DAT`
Statistical information about HiRDB files for database manipulation	Each synchronization point	`fil_DAT`
Deferred write processing statistical information	Each activation of deferred write daemon	`dfw_DAT`

Edited information	Output unit	File name
Static optimization information	Preprocessing of one SQL	`sop_DAT`
Dynamic optimization information	Each server within an SQL	`dop_DAT`
SQL object execution information	Each parallel SQL	`pcd_DAT`
SQL object transfer statistical information	Each parallel SQL	`obj_DAT`
SQL statement statistical information	Execution of one SQL	`sqh_DAT`
`CONNECT`/`DISCONNECT` statistical information	Each `CONNECT`/`DISCONNECT`	`cnc_DAT`
Foreign server operation statistical information	Each transaction	`fsv_DAT`
Foreign server utilization statistical information	Each thread execution	`hba_DAT`

Notes

1. Each file has a predefined name. If the same file name is found in the user-specified directory, the existing file may be overwritten. To avoid this, the existing file should be renamed or a different directory should be specified.
2. A DAT-format file cannot store more than 2 gigabytes of data. When you are using a large number of system log files, make sure that the size of the output data does not exceed 2 gigabytes by specifying appropriate options to narrow the data to be output.
3. The system activity statistical information consists of too many columns to be imported to Excel as is. Use the `cut` command to delete unneeded columns before importing the data.

   The following is an example:

   ```
   cut -d "," -f1-256 /tmp/sys_DAT > /tmp/sys_DAT.out
   ```

14.4.3 Record formats of DAT-format files

Tables 14-5 through 14-22 show the record formats of the DAT-format files.

The following notes apply to these tables:

- The title bars shown in the tables are displayed when the `-b` option is specified. For system activity statistical information, three title bar lines are displayed at the beginning of the file; in the case of all other statistical information types, one title bar line is displayed at the beginning of the file.

- The length of the title bar for each field is not included in the maximum length displayed in the tables. You should take this into account when you process data in DAT-format files that includes the title bars.
- In the *Remarks* column, — means that there are no remarks.
- If a data item's attribute is character string, the maximum length of the data does not include the double quotation marks (`""`) that enclose the character string data. If the utility detects uneditable data for an item whose attribute is numeric, it displays 10 asterisks (`**********`), which may exceed the maximum length (of 3, 5, or 6 bytes).

Table 14-5: Record format of a DAT-format file (system activity statistical information (1))

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	—
2	Server name (`SERVER`)		8	`********` for overall statistical information
3	Edit time (`START`)		11	Format *MM/DD/ hh:mm*

No.	Field name (title bar)			Attribute	Maximum length	Remarks
4	Schedule information (SCHEDULE)	Length of schedule queue (QUEUE LEN(SCH))	Number of occurrences (FREQ)	Numeric	10	—
5			Maximum value (MAX)			
6			Minimum value (MIN)			
7			Average value (AVG)			
8		Scheduled message length (MESSAGE LEN)	Number of occurrences (FREQ)			—
9			Maximum value (MAX)			In bytes
10			Minimum value (MIN)			
11			Average value (AVG)			
12	Process information (PROCESS)	Server abnormal terminations count (SERVER ABORT)	Number of occurrences (FREQ)	Numeric	10	Processing information continues with Nos. 174-177.
13		Server abnormal terminations count for HiRDB internal server (SYSTEM SERVER ABORT)	Number of occurrences (FREQ)			—
14		Number of processes (PROCESS COUNT)	Maximum value (MAX)			
15			Minimum value (MIN)			
16			Average value (AVG)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
17	Transaction information (TRANSACTION)	Commits count (COMMIT)	Number of occurrences (FREQ)	Numeric	10	—
18		Rollbacks count (ROLLBACK)	Number of occurrences (FREQ)			
19	Name server information (NAME)	System-specific information (CACHE HIT)	Number of occurrences (FREQ)	Numeric	10	—
20		System-specific information (LOCAL HIT)	Number of occurrences (FREQ)			
21		System-specific information (LOOK UP)	Number of occurrences (FREQ)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
22	RPC information (RPC)	System-specific information (RESPONSE TIME)	Number of occurrences (FREQ)	Numeric	10	RPC information for users is listed in Nos. 178-199 and 252-259.
23			Maximum value (MAX)			—
24			Minimum value (MIN)			
25			Average value (AVG)			
26		System-specific information (USER SERVICE TIME)	Number of occurrences (FREQ)			—
27			Maximum value (MAX)			
28			Minimum value (MIN)			
29			Average value (AVG)			
30		System-specific information (TIME OUT)	Number of occurrences (FREQ)			
31		System-specific information (ERROR)	Number of occurrences (FREQ)			
32		System-specific information (SEND)	Number of occurrences (FREQ)			
33		System-specific information (RECEIVE)	Number of occurrences (FREQ)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
34	Lock information (LOCK)	Lock release wait time (WAIT TIME)	Number of occurrences (FREQ)	Numeric	10	—
35			Maximum value (MAX)			In millisecond s
36			Minimum value (MIN)			
37			Average value (AVG)			
38		Lock queue length (number of users) (QUEUE LEN)	Number of occurrences (FREQ)			—
39			Maximum value (MAX)			
40			Minimum value (MIN)			
41			Average value (AVG)			
42		Deadlocks count (DEADLOCK)	Number of occurrences (FREQ)			
43		Utilization factor of locked resources management table (USE LOCK TABLE)	Number of occurrences (FREQ)			
44			Maximum value (MAX)		3	Percentage
45			Minimum value (MIN)			
46			Average value (AVG)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
47	Shared memory information (`SHARED MEMORY`)	System-specific information (`STATIC GET SIZE`)	Number of occurrences (`FREQ`)	Numeric	10	—
48			Maximum value (`MAX`)			In bytes
49			Minimum value (`MIN`)			
50			Average value (`AVG`)			
51		System-specific information (`STATIC POOL SIZE`)	Number of occurrences (`FREQ`)			—
52			Maximum value (`MAX`)			In bytes
53			Minimum value (`MIN`)			
54			Average value (`AVG`)			
55		System-specific information (`DYNAMIC GET SIZE`)	Number of occurrences (`FREQ`)			—
56			Maximum value (`MAX`)			In bytes
57			Minimum value (`MIN`)			
58			Average value (`AVG`)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
59		System-specific information (DYNAMIC POOL SIZE)	Number of occurrences (FREQ)			—
60			Maximum value (MAX)			In bytes
61			Minimum value (MIN)			
62			Average value (AVG)			
63		Size of shared memory allocated for server or for HiRDB internal server (SEGMENT SIZE)	Average value (AVG)			In KB
64		Shared static memory allocation size (STATIC SIZE)	Average value (AVG)			
65		Shared dynamic memory allocation size (DYNAMIC SIZE)	Average value (AVG)			
66		Shared memory allocation size for global buffer (SIZE FOR BUFFER)	Average value (AVG)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
67	Synchronization point information (SYNC POINT)	Synchronization point dump collection time (GET INTERVAL TIME)	Number of occurrences (FREQ)	Numeric	10	—
68			Maximum value (MAX)			In milliseconds
69			Minimum value (MIN)			
70			Average value (AVG)			
71		Synchronization point dump interval (GET TIME)	Number of occurrences (FREQ)			—
72			Maximum value (MAX)			In milliseconds
73			Minimum value (MIN)			
74			Average value (AVG)			
75	Log information (LOG)	Buffer-fulls count (BUFFER FULL)	Number of occurrences (FREQ)	Numeric	10	—
76		Waits count due to no current buffer (WAIT THREAD)	Number of occurrences (FREQ)			
77		Output block length (OUTPUT BLOCK LEN)	Number of occurrences (FREQ)			
78			Maximum value (MAX)			In bytes
79			Minimum value (MIN)			
80			Average value (AVG)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
81		Non-bus data length (`NOT BUS LEN`)	Number of occurrences (`FREQ`)			—
82			Maximum value (`MAX`)			In bytes
83			Minimum value (`MIN`)			
84			Average value (`AVG`)			
85		Number of buffer sectors waiting for I/O (`WAIT BUFFER FOR IO`)	Number of occurrences (`FREQ`)			—
86			Maximum value (`MAX`)			
87			Minimum value (`MIN`)			
88			Average value (`AVG`)			
89		Number of file write operations (`WRITE TO FILE`)	Number of occurrences (`FREQ`)			—
90		Write errors count (`WRITE ERROR`)	Number of occurrences (`FREQ`)			
91		Log file swapping time (`LOG FILE SWAP TIME`)	Number of occurrences (`FREQ`)			
92			Maximum value (`MAX`)			In millisecond s
93			Minimum value (`MIN`)			
94			Average value (`AVG`)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
95		Log input data length (LOG INPUT DATA)	Number of occurrences (FREQ)			—
96			Maximum value (MAX)			In bytes
97			Minimum value (MIN)			
98			Average value (AVG)			
99		Number of file read operations (READ FROM FILE)	Number of occurrences (FREQ)			—
100		Read errors count (READ ERROR)	Number of occurrences (FREQ)			—

Table 14-6: Record format of a DAT-format file (system activity statistical information (2))

No.	Field name (title bar)			Attribute	Maximum length	Remarks
101	Dictionary information (DICTIONARY)	Number of table definition information acquisition requests (TBL-DEF GET REQ)	Number of occurrences (FREQ)	Numeric	10	Dictionary information is continued to Nos. 200-251 and 260-267.
102		Table definition information buffer hits count (TABLE CACHE HIT)	Number of occurrences (FREQ)			—
103		Number of definition information items in table definition information buffer (CASHED TBL-DEF)	Number of occurrences (FREQ)			—
104			Maximum value (MAX)			
105			Minimum value (MIN)			
106			Average value (AVG)			
107		Size of table definition information buffer used per table definition information item (USE TBL-DEF SIZE)	Number of occurrences (FREQ)			—
108			Maximum value (MAX)			In bytes
109			Minimum value (MIN)			
110			Average value (AVG)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
111		Size of area used by table definition information buffer (`CACHED TBL-DEF SIZE`)	Number of occurrences (`FREQ`)			—
112			Maximum value (`MAX`)			In bytes
113			Minimum value (`MIN`)			
114			Average value (`AVG`)			
115		Table access privilege information acquisitions count (`ACCESS PRIV CHECK`)	Number of occurrences (`FREQ`)			—
116		Table access privilege information buffer hits count (`CACHE HIT(ACCESS)`)	Number of occurrences (`FREQ`)			—
117		User privilege information acquisition requests (`CON/DBA DEF GET REQ`)	Number of occurrences (`FREQ`)			—
118		User privilege information buffer hits count (`CON/DBA CACHE HIT`)	Number of occurrences (`FREQ`)			—
119		Number of users using the user privilege information buffer (`CON/DBA CACHED USER`)	Number of occurrences (`FREQ`)			—
120			Maximum value (`MAX`)			—
121			Minimum value (`MIN`)			—
122			Average value (`AVG`)			—

No.	Field name (title bar)			Attribute	Maximum length	Remarks
123		Length of communication with dictionary server (`DIC TRANS DATA LEN`)	Number of occurrences (`FREQ`)			—
124			Maximum value (`MAX`)			In bytes
125			Minimum value (`MIN`)			
126			Average value (`AVG`)			
127		Dictionary server communications count (`TRANS`)	Number of occurrences (`FREQ`)			—
128		Number of view analysis information acquisition requests (`VIEW DEF GET REQ`)	Number of occurrences (`FREQ`)			
129		View analysis information buffer hits count (`VIEW DEF CACHE HIT`)	Number of occurrences (`FREQ`)			
130		Number of view analysis information items in view analysis information buffer (`VIEW CACHED DEF`)	Number of occurrences (`FREQ`)			—
131		Size of view analysis information buffer used per view analysis information item (`USED VIEW SIZE`)	Number of occurrences (`FREQ`)			
132			Maximum value (`MAX`)			In bytes
133			Minimum value (`MIN`)			
134			Average value (`AVG`)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
135		Length of view analysis information buffer used (`VIEW CACHE SIZE`)	Number of occurrences (`FREQ`)			—
136			Maximum value (`MAX`)			In bytes
137			Minimum value (`MIN`)			
138			Average value (`AVG`)			
139		Length of view analysis information resulting in buffer miss (`CACHE MISS VIEW SIZE`)	Number of occurrences (`FREQ`)			—
140			Maximum value (`MAX`)			In bytes
141			Minimum value (`MIN`)			
142			Average value (`AVG`)			
143	SQL object information for single server, front-end server, back-end server, or dictionary server (`FES-BES-DIC (SDS) INFORMATION`)	Number of SQL object acquisition requests (`SQLOBJ GET REQ`)	Number of occurrences (`FREQ`)			—
144		SQL object buffer hits count (`SQLOBJ CACHE HIT`)	Number of occurrences (`FREQ`)			
145		Number of SQL objects in SQL object buffer (`CACHED SQLOBJ`)	Number of occurrences (`FREQ`)			
146			Maximum value (`MAX`)			
147			Minimum value (`MIN`)			
148			Average value (`AVG`)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
149		Total length of SQL objects in SQL object buffer (`CACHED SQLOBJ SIZE`)	Number of occurrences (`FREQ`)			—
150			Maximum value (`MAX`)			In KB
151			Minimum value (`MIN`)			
152			Average value (`AVG`)			
153		Number of SQL objects swapped out of SQL object buffer (`SWAP OUT SQLOBJ`)	Number of occurrences (`FREQ`)			—
154		SQL object length (`SQLOBJ LEN`)	Number of occurrences (`FREQ`)			—
155			Maximum value (`MAX`)			In bytes
156			Minimum value (`MIN`)			
157			Average value (`AVG`)			
158		Number of acquisition requests for stored procedure objects (`STROBJ GET REQ`)	Number of occurrences (`FREQ`)			—
159		SQL object buffer hits count for stored procedure objects (`STROBJ CACHE HIT`)	Number of occurrences (`FREQ`)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
160		Number of stored procedure objects in SQL object buffer (`CACHED STROBJ`)	Number of occurrences (`FREQ`)			—
161			Maximum value (`MAX`)			
162			Minimum value (`MIN`)			
163			Average value (`AVG`)			
164		Total length of stored procedure objects in SQL object buffer (`CACHED STROBJ SIZE`)	Number of occurrences (`FREQ`)			—
165			Maximum value (`MAX`)			In KB
166			Minimum value (`MIN`)			
167			Average value (`AVG`)			
168		Number of stored procedure objects swapped out of SQL object buffer (`SWAP OUT STROBJ`)	Number of occurrences (`FREQ`)			—
169		Stored procedure object length (`STROBJ LEN`)	Number of occurrences (`FREQ`)			
170			Maximum value (`MAX`)			In bytes
171			Minimum value (`MIN`)			
172			Average value (`AVG`)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
173		Number of times stored procedure objects were recompiled (STROBJ RECOMPILE)	Number of occurrences (FREQ)			—
174	Process information (PROCESS)	Number of server processes under service execution (SERVICE COUNT)	Maximum value (MAX)	Numeric	10	—
175			Minimum value (MIN)			
176			Average value (AVG)			
177		Number of service requests exceeding maximum number of startup processes (REQUEST OVER)	Number of requests (FREQ)			
178	RPC information (RPC)	Service response time for local unit's servers (RESPONSE ON OWN UNIT)	Number of occurrences (FREQ)	Numeric	10	—
179			Maximum value (MAX)			In 100 microseconds
180			Minimum value (MIN)			
181			Average value (AVG)			
182		Service response time for remote unit's servers (RESPONSE TO OTHER UNIT)	Number of occurrences (FREQ)			—
183			Maximum value (MAX)			In 100 microseconds
184			Minimum value (MIN)			
185			Average value (AVG)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
186		Execution time per service from local unit's servers (`EXEC TIME ON OWN UNIT`)	Number of occurrences (`FREQ`)			—
187			Maximum value (`MAX`)			In 100 microseconds
188			Minimum value (`MIN`)			
189			Average value (`AVG`)			
190		Execution time per service from remote unit's servers (`EXEC TIME FROM OTHER UNIT`)	Number of occurrences (`FREQ`)			—
191			Maximum value (`MAX`)			In 100 microseconds
192			Minimum value (`MIN`)			
193			Average value (`AVG`)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
194		Number of SENDs to local process (SEND TO OWN PRCS)	Number of occurrences (FREQ)			—
195		Number of SENDs to other processes on local unit (SEND TO OTHER PRCS)	Number of occurrences (FREQ)			
196		Number of SENDs to remote unit (SEND TO OTHER UNIT)	Number of occurrences (FREQ)			
197		Number of RECEIVEs from local process (RECEIVE FROM OWN PRCS)	Number of occurrences (FREQ)			
198		Number of RECEIVEs from other processes on local unit (RECEIVE FROM OTHER PRCS)	Number of occurrences (FREQ)			
199		Number of RECEIVEs from remote unit (RECEIVE FROM OTHER UNIT)	Number of occurrences (FREQ)			

Table 14-7: Record format of a DAT-format file (system activity statistical information (3))

No.	Field name (title bar)			Attribute	Maximum length	Remarks
200	Dictionary information (`DICTIONARY`)	Number of type definition information acquisition requests (`TYPE-DEF GET REQ`)	Number of occurrences (`FREQ`)	Numeric	10	—
201		User-defined type information buffer hits count (`TYPE-DEF CACHE HIT`)	Number of occurrences (`FREQ`)			
202		Number of type definition information items in user-defined type information buffer (`CACHED TYPE-DEF`)	Number of occurrences (`FREQ`)			
203			Maximum value (`MAX`)			
204			Minimum value (`MIN`)			
205			Average value (`AVG`)			
206		Size of area used by user-defined type information buffer per type definition information item (`TYPE-DEF CACHE SIZE`)	Number of occurrences (`FREQ`)			
207			Maximum value (`MAX`)			In bytes
208			Minimum value (`MIN`)			
209			Average value (`AVG`)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
210		Total size of area used by user-defined type information buffer (`TYPE-DEF CACHE TOTAL SIZE`)	Number of occurrences (`FREQ`)			—
211			Maximum value (`MAX`)			In bytes
212			Minimum value (`MIN`)			
213			Average value (`AVG`)			
214		Size of user-defined type information buffer allocated (`TYPE-DEF CACHE ALLOC SIZE`)	Number of occurrences (`FREQ`)			—
215			Maximum value (`MAX`)			In bytes
216			Minimum value (`MIN`)			
217			Average value (`AVG`)			
218		Number of routine definition information acquisition requests (`RTN-DEF GET REQ`)	Number of occurrences (`FREQ`)			—

No.	Field name (title bar)			Attribute	Maximum length	Remarks
219		Routine definition information buffer hits count (`RTN-DEF CACHE HIT`)	Number of occurrences (`FREQ`)			—
220		Number of routine definition information items in routine definition information buffer (`CACHED RTN-DEF`)	Number of occurrences (`FREQ`)			
221			Maximum value (`MAX`)			
222			Minimum value (`MIN`)			
223			Average value (`AVG`)			
224		Size of area used by routine definition information buffer per routine definition information item (`RTN-DEF CACHE SIZE`)	Number of occurrences (`FREQ`)			—
225			Maximum value (`MAX`)			In bytes
226			Minimum value (`MIN`)			
227			Average value (`AVG`)			
228		Total size of area used by routine definition information buffer (`RTN-DEF CACHE TOTAL SIZE`)	Number of occurrences (`FREQ`)			—
229			Maximum value (`MAX`)			In bytes
230			Minimum value (`MIN`)			
231			Average value (`AVG`)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
232		Size of routine definition information buffer allocated (RTN-DEF CACHE ALLOC SIZE)	Number of occurrences (FREQ)			—
233			Maximum value (MAX)			In bytes
234			Minimum value (MIN)			
235			Average value (AVG)			
236		Number of routine definition acquisition requests for plug-in-provided functions (PLG-RTN GET REQ)	Number of occurrences (FREQ)			—
237		Routine definition information buffer hits count for plug-in provided functions (PLG-RTN CACHE HIT)	Number of occurrences (FREQ)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
238		Number of registry information acquisition requests (`REGISTRY GET REQ`)	Number of occurrences (`FREQ`)			—
239		Registry information buffer hits count (`REGISTRY CACHE HIT`)	Number of occurrences (`FREQ`)			
240		Number of registry information items in registry information buffer (`CACHED REGISTRY-DEF`)	Number of occurrences (`FREQ`)			
241			Maximum value (`MAX`)			
242			Minimum value (`MIN`)			
243			Average value (`AVG`)			
244		Size of registry information buffer per registry information item (`REGISTRY CACHE SIZE`)	Number of occurrences (`FREQ`)			
245			Maximum value (`MAX`)			In bytes
246			Minimum value (`MIN`)			
247			Average value (`AVG`)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
248		Total size of area used by registry information buffer (`REGISTRY CACHE TOTAL SIZE`)	Number of occurrences (`FREQ`)			—
249			Maximum value (`MAX`)			In bytes
250			Minimum value (`MIN`)			
251			Average value (`AVG`)			
252	RPC information (`RPC`)	Number of HiRDB-reserved ports used (`REGISTERED PORTS`)	Number of occurrences (`FREQ`)	Numeric	10	—
253			Maximum value (`MAX`)			
254			Minimum value (`MIN`)			
255			Average value (`AVG`)			
256		Number of ports automatically assigned by OS when HiRDB-reserved ports ran out (`ASSIGNED PORTS`)	Number of occurrences (`FREQ`)			
257			Maximum value (`MAX`)			
258			Minimum value (`MIN`)			
259			Average value (`AVG`)			

No.	Field name (title bar)			Attribute	Maximum length	Remarks
260	Dictionary information (DICTIONARY)	Directory-registered user authentication time (DIRECTORY USER CHECK TIME)	Number of occurrences (FREQ)	Numeric	10	—
261			Maximum value (MAX)			In microseconds
262			Minimum value (MIN)			
263			Average value (AVG)			
264		Group checking time (GROUP CHECK TIME)	Number of occurrences (FREQ)			—
265			Maximum value (MAX)			In microseconds
266			Minimum value (MIN)			
267			Average value (AVG)			

Table 14-8: Record format of a DAT-format file: UAP statistical information

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	—
2	UAP name (`UAP NAME`)			
3	UAP or transaction execution start time (`START`)	Character	14	Format: *MM/DD/ hh:mm:ss*
4	No. 3 in microseconds (`START (MICRO)`)	Numeric	6	Value in seconds is not included.
5	UAP or transaction execution termination time (`END`)	Character	14	Format: *MM/DD/ hh:mm:ss*
6	No. 5 in microseconds (`END (MICRO)`)	Numeric	6	Value in seconds is not included.
7	Termination status (`STATUS`)	Character	1	`N`: Normal termination `E`: Termination after rollback
8	UAP or transaction execution time (`UAP EXEC TIME(S)`)	Numeric	10	In seconds (value less than a second is rounded)
9	Size of table definition information buffer used (`CACHE SIZE`)			In bytes
10	Number of SQL object acquisition requests (`REQ (SQL)`)			—
11	SQL object buffer hits count (`HITS (SQL)`)			
12	SQL object creations count (`CREATE`)			
13	Maximum size of SQL object created (`SQL MAX`)			In bytes

No.	Field name (title bar)	Attribute	Maximum length	Remarks
14	`COMMIT` statement executions count (`COMMIT`)			--
15	`ROLLBACK` statement executions count (`ROLLBACK`)			
16	Number of rows actually retrieved by `FETCH` or `SELECT` statement (`FETCH ROW`)			
17	Number of rows actually deleted by `DELETE` statement (`DELETE ROW`)			
18	Number of rows actually inserted by `INSERT` statement (`INSERT ROW`)			
19	Number of rows actually updated by `UPDATE` statement (`UPDATE ROW`)			
20	SQL statement preprocessings count (`SET`)			
21	`OPEN` statement executions count (`OPEN`)			
22	`FETCH` statement executions count (`FETCH`)			
23	`CLOSE` statement executions count (`CLOSE`)			
24	`DESCRIBE` statement executions count (`DESCRIBE`)			
25	`SELECT` statement executions count (`SELECT`)			
26	`INSERT` statement executions count (`INSERT`)			
27	`UPDATE` statement executions count (`UPDATE`)			
28	`DELETE` statement executions count (`DELETE`)			

No.	Field name (title bar)	Attribute	Maximum length	Remarks
29	`LOCK` statement executions count (`LOCK`)			—
30	`CREATE TABLE` executions count (`CREATE TABLE`)			
31	`ALTER TABLE` executions count (`ALTER TABLE`)			
32	`DROP TABLE` executions count (`DROP TABLE`)			
33	`CREATE INDEX` executions count (`CREATE INDEX`)			
34	`DROP INDEX` executions count (`DROP INDEX`)			
35	`COMMENT TABLE` executions count (`COMMENT TABLE`)			
36	`COMMENT COLUMN` executions count (`COMMENT COLUMN`)			
37	`CREATE SCHEMA` executions count (`CREATE SCHEMA`)			
38	`DROP SCHEMA` executions count (`DROP SCHEMA`)			
39	`GRANT RDAREA` executions count (`GRANT RDAREA`)			
40	`GRANT SCHEMA` executions count (`GRANT SCHEMA`)			
41	`GRANT` access-privilege executions count (`GRANT ACCESS`)			
42	`GRANT CONNECT` executions count (`GRANT CONNECT`)			
43	`GRANT DBA` executions count (`GRANT DBA`)			
44	`REVOKE RDAREA` executions count (`REVOKE RDAREA`)			
45	`REVOKE SCHEMA` executions count (`REVOKE SCHEMA`)			—
46	`REVOKE` access-privilege executions count (`REVOKE ACCESS`)			
47	`REVOKE CONNECT` executions count (`REVOKE CONNECT`)			

No.	Field name (title bar)	Attribute	Maximum length	Remarks
48	REVOKE DBA executions count (REVOKE DBA)			
49	CREATE VIEW executions count (CREATE VIEW)			
50	DROP VIEW executions count (DROP VIEW)			
51	System-specific information (UNUSED)			
52	System-specific information (UNUSED)			
53	PURGE TABLE statement executions count (PURGE TABLE)			
54	Other executions count (MISC)			
55	Number of acquisition requests for stored procedure objects (SQLOBJ GET REQ)			
56	SQL object buffer hits count for stored procedure objects (SQLOBJ CACHE HIT)			
57	CREATE PROCEDURE executions count (CREATE PROCEDURE)			
58	DROP PROCEDURE executions count (DROP PROCEDURE)			
59	CALL statement executions count (CALL)			
60	DESCRIBE statement (INPUT) executions count (DESCRIBE INPUT)			
61	ALTER PROCEDURE executions count (ALTER PROCEDURE)			
62	UAP or transaction execution time (UAP EXEC TIME (MS))			In milliseconds (includes a value in seconds; a value less than a millisecond is rounded)
63	Service name (SERVICE NAME)	Character	31	—
64	Lock release wait time (LOCK TIME)[1]	Numeric	10	In seconds
65	No. 64 in microseconds (LOCK TIME (MICRO))[1]		6	Value in seconds is not included.

No.	Field name (title bar)	Attribute	Maximum length	Remarks
66	Output timing for the corresponding row (`OUTPUT TIMING`)	Character	1	`C`: In UAP executions `T`: In transactions
67	Front-end server name (`FES NAME`)		8	--
68	Connect number (`CONNECT NO`)	Numeric	10	Sequence number assigned to each connection within one front-end server
69	Font-end server process ID (`FES PROCESS ID`)			--
70	CPU time (`CPU TIME`)4			In milliseconds
71	Single server or front-end server processing time (`SERV EXEC TIME`)2			In seconds
72	No. 71 in microseconds (`SERV EXEC TIME (MICRO)`)		6	Value in seconds is not included.
73	Total input/output time (`IO TIME`)5		10	In seconds
74	No. 73 in microseconds (`IO TIME (MICRO)`)5		6	Value in seconds is not included.
75	Maximum input/output time (`MAX IO TIME`)3		10	In seconds
76	No. 75 in microseconds (`MAX IO TIME (MICRO)`)		6	Value in seconds is not included.
77	Minimum input/output time (`MIN IO TIME`)3		10	In seconds
78	No. 77 in microseconds (`MIN IO TIME (MICRO)`)		6	Value in seconds is not included.
79	Reference operations count on data page, index page, or directory page (`DID RF CNT`)5		10	--
80	Update operations count on data page, index page, or directory page (`DID UPD CNT`)5			--
81	Buffer hits count for data page, index page, or directory page (`DID HIT CNT`)5			--
82	Buffer hit ratio for data page, index page, or directory page (`DID BUF HIT RATIO`)		3	Percentage

No.	Field name (title bar)	Attribute	Maximum length	Remarks
83	Real READs count for data page, index page, or directory page (DID READ CNT)[5]		10	—
84	Real WRITEs count for data page, index page, or directory page (DID WRITE CNT)[5]			—
85	Reference operations count on LOB column data page (LOB RF CNT)[5]			—
86	Update operations count on LOB column data page (LOB UPD CNT)[5]			—
87	Reference buffer hits count for LOB column data page (LOB RF HIT CNT)[5]			—
88	Update buffer hits count for LOB column data page (LOB UPD HIT CNT)[5]			—
89	Reference buffer hits count for LOB column data page (LOB RF BUF HIT RATIO)		3	Percentage
90	Update buffer hits count for LOB column data page (LOB UPD BUF HIT RATIO)			
91	Real READs count for LOB column database (LOB READ CNT)[5]		10	—
92	Real WRITEs count for LOB column database (LOB WRITE CNT)[5]			—
93	Global buffer flashes count (BUF FLASH CNT)[5]			—
94	READ waits count for global buffer (BUF READ WAIT CNT)[5]			—
95	WRITE waits count for global buffer (BUF WRITE WAIT CNT)[5]			—
96	Global buffer lock release waits count (BUF LOCK WAIT CNT)[5]			—
97	Maximum number of work table files (MAX WKFILE NUM)			—
98	Maximum number of work table file extensions (MAX WKFILE EXP CNT)			—

No.	Field name (title bar)	Attribute	Maximum length	Remarks
99	Maximum size of work table file (`MAX WKFILE VOLUME`)			In MB
100	Work table file `READ`s count (`WKFILE READ CNT`)5			—
101	Work table file `WRITE`s count (`WKFILE WRITE CNT`)5			—
102	Number of forced outputs on work table buffer (`WKFILE BUF FORCED OUT CNT`)5			—
103	Maximum hash table size in batch processing mode (`MAX WHOLE HASH TABLE SIZE`)			In KB
104	Maximum level-1 bucket size (`MAX BUCKET SIZE LEVEL 1`)			
105	Maximum level-2 bucket size (`MAX BUCKET SIZE LEVEL 2`)			
106	Maximum level-3 bucket size (`MAX BUCKET SIZE LEVEL 3`)			

No.	Field name (title bar)	Attribute	Maximum length	Remarks
107	Number of times page search for reutilization of free space failed (`REUSE FAILURE CNT`)[5]			—
108	Number of times the mode changed from *new page allocate* to *free page reuse* on a table with `SEGMENT REUSE` specified (`REUSE CHANGE CNT`)[5]			
109	Number of times data pages and index pages were referenced using the local buffer (`LOCAL BUF RF CNT`)[5]			
110	Number of times data pages and index pages were updated using the local buffer (`LOCAL BUF UPD CNT`)[5]			
111	Local buffer hit count for data pages and index pages (`LOCAL BUF HIT CNT`)[5]			
112	Number of real `READ`s on data pages and index pages using the local buffer (`LOCAL BUF READ CNT`)[5]			
113	Number of real `WRITE`s on data pages and index pages using the local buffer (`LOCAL BUF WRITE CNT`)[5]			
114	Local buffer flush count (`LOCAL BUF FLUSH CNT`)[5]			
115	Number of asynchronous `READ` requests (`AR REQ CNT`)[5]			
116	Number of times synchronization waits occurred during asynchronous `READ` (`AR SYNC WAIT CNT`)[5]			
117	Total synchronization wait time during asynchronous `READ` (`AR SYNC WAIT TIME TOTAL`)[5]			In seconds
118	No. 177 in microseconds (`AR SYNC WAIT TIME TOTAL (MICRO)`)[5]		6	Value in seconds is not included.
119	Average synchronization wait time during asynchronous `READ` (`AR SYNC WAIT TIME AVERAGE`)		10	In seconds

No.	Field name (title bar)	Attribute	Maximum length	Remarks
120	No. 119 in microseconds (`AR SYNC WAIT TIME AVERAGE (MICRO)`)		6	Value in seconds is not included.
121	Total synchronous input/output time during asynchronous `READ` (`AR SYNC IO TIME TOTAL`)[5]		10	In seconds
122	No. 121 in microseconds (`AR SYNC IO TIME TOTAL (MICRO)`)[5]		6	Value in seconds is not included.
123	Average synchronous input/output time during asynchronous `READ` (`AR SYNC IO TIME AVERAGE`)		10	In seconds
124	No. 123 in microseconds (`AR SYNC IO TIME AVERAGE (MICRO)`)		6	Value in seconds is not included.
125	Maximum comparison count when a hash row partitioning table was searched during hash join, subquery hash execution (`MAX HASH COMP COUNT`)[3]		10	--
126	Total comparison count when a hash row partitioning table was searched during hash join, subquery hash execution (`HASH COMP COUNT`)[5]		10	--
127	Total hash row partitioning table search count during hash join, subquery hash execution (`HASH HIT COUNT`)[5]		10	--

Note

For a HiRDB/Parallel Server, Nos. 73, 74, 79-102, 126, and 127 indicate the total value for all back-end servers.

[1] For a HiRDB/Parallel Server, this is the total lock release wait time (including parallel processing) resulting in the transaction.

[2] If the `pdstbegin` command is executed while being connected, or transaction processing is underway, statistical information is collected for the next connection or transaction, not the current connection or transaction.

[3] For a HiRDB/Parallel Server, the value is for all back-end servers.

[4] This is the total CPU time for all servers in the transaction.

[5] If the information is output for each transaction, this is the cumulative `CONNECT` value.

Table 14-9: Record format of a DAT-format file (SQL statistical information)

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	—
2	UAP name (`UAP NAME`)			
3	SQL execution start time (`START`)		14	Format: *MM/DD/hh:mm:ss*
4	No. 3 in microseconds (`START (MICRO)`)	Numeric	6	Value in seconds is not included.
5	SQL execution termination time (`END`)	Character	14	Format: *MM/DD/hh:mm:ss*
6	No. 5 in microseconds (`END (MICRO)`)	Numeric	6	Value in seconds is not included.
7	Length of created SQL object (`PDATA`)		10	In bytes
8	Number of back-end servers that sent SQL object (`BES COUNT`)			—
9	SQL execution time (`SQL EXEC TIME (MICRO)`)			In microseconds (includes the value in seconds)
10	Number of rows retrieved (`FETCH ROW`)			—
11	Number of rows inserted (`INSERT ROW`)			
12	Number of rows updated (`UPDATE ROW`)			
13	Number of rows deleted (`DELETE ROW`)			
14	Number of rows created in work table (`CREATE LIST`)			
15	Number of rows deleted from work table (`DROP LIST`)			
16	SQL execution time (`SQL EXEC TIME (SEC)`)			In seconds (value is rounded off)
17	Service name (`SERVICE NAME`)	Character	31	—
18	Front-end server name (`FES NAME`)		8	

No.	Field name (title bar)	Attribute	Maximum length	Remarks
19	Connection number (`CONNECT NO`)	Numeric	10	Serial number assigned to each connection in one front-end server.
20	SQL counter (`SQL COUNTER`)			—
21	Front-end server process ID (`FES PROCESS ID`)			
22	Operation code (`OPERATION CODE`)[1]	Character	4	—
23	Section number (`SECTION NO`)	Numeric	5	
24	SQL type code (`SQL TYPE CODE`)[2]	Character	4	
25	SQL code (`SQLCODE`)	Numeric	10	
26	Server name (`SERVER NAME`)	Character	8	
27	Number of times processing requests were issued to the foreign server (`FOREIGN REQUEST COUNT`)	Numeric	10	—
28	Foreign server processing time (`FOREIGN EXEC TIME`)			In seconds
29	No. 28 in microseconds (`FOREIGN EXEC TIME(MICRO)`)			In microseconds (includes the value in seconds)
30	Number of rows retrieved on the foreign server (`FOREIGN FETCH ROW`)			—
31	Total comparison count when a hash row partitioning table was searched during hash join, subquery hash execution (`HASH COMP COUNT`)	Numeric	10	—
32	Total hash row partitioning table search count during hash join, subquery hash execution (`HASH HIT COUNT`)	Numeric	10	—
33	Number of rows inserted in the foreign server (`FOREIGN INSERT ROW`)	Numeric	10	—
34	Number of rows updated on the foreign server (`FOREIGN UPDATE ROW`)	Numeric	10	—
35	Number of rows deleted on the foreign server (`FOREIGN DELETE ROW`)	Numeric	10	—

Note

Nos. 27-30 and 33-35 are output when the HiRDB External Data Access facility is used. If this facility is not used, `0` or `****` is output.

1 See Table 14-10 for the operation codes.

2 See Table 14-11 for the SQL type codes.

Table 14-10: Operation codes

Operation code	Corresponding SQL statements
AUI2	DELETE statement (static SQL), INSERT statement (static SQL), UPDATE statement (static SQL), LOCK statement (static SQL), PURGE TABLE statement (static SQL), single-row SELECT statement (static SQL), FREE LOCATOR statement (static SQL)
AUI3	Assignment statement (static SQL)
AUX	EXECUTE statement
AUXI	EXECUTE IMMEDIATE statement and all definition SQL statements
AUXO	EXECUTE statement (INTO specified)
CALL	CALL statement
CLOS	CLOSE statement
CMIT	COMMIT statement
CNCT	CONNECT statement
CPRP	Commit Prepare*
DESC	DESCRIBE statement (OUTPUT specified)
DEST	DESCRIBE TYPE statement
DISC	DISCONNECT statement, COMMIT statement (RELEASE specified)
DISR	ROLLBACK statement (RELEASE specified)
DIST	Disconnect + Tran Check*
DSCM	Used by system
DSPR	Used by system
DSRL	Used by system
FETC	FETCH statement
GETD	GET DIAGNOSTICS
HVAR	DESCRIBE statement (INPUT specified)
JARI	INSTALL JAR
JARR	REPLACE JAR
JARU	REMOVE JAR
OPEN	OPEN statement (dynamic SQL)

Operation code	Corresponding SQL statements
OPN2	OPEN statement (static SQL)
OPNR	OPEN statement (dynamic SQL (multiple cursors))
RENV	Used by system
RNCN	CONNECT statement (TO specified)
RNDS	DISCONNECT statement (TO specified)
RNSC	SET CONNECTION statement
ROLL	ROLLBACK statement
RSDC	DESCRIBE statement (OUTPUT, RESULT SET specified)
RSFT	FETCH statement (RESULT SET specified)
RSCL	CLOSE statement (RESULT SET specified)
SAUH	SET SESSION AUTHORIZATION statement
SET	PREPARE statement
SINF	Used by system
SOPT	Used by system
SVLS	Used by system
THRE	Used by system
THSU	Used by system
TRCK	Used by system
TRC2	Used by system
TRST	Used by system
TSCM	Used by system
TSRL	Transfer Rollback*
TSPR	Transfer Prepare*
ALCR	ALLOCATE CURSOR statement
DSET	DEALLOCATE PREPARE statement

* Output only when the XA interface is used.

Table 14-11: SQL type codes

No.	SQL statement	SQL type code (hexadecimal)
1	`CREATE TABLE`	`1038`
2	`DROP TABLE`	`103C`
3	`ALTER TABLE`	`1040`
4	`CREATE INDEX`	`1044`
5	`DROP INDEX`	`1048`
6	`COMMENT ON TABLE`	`104C`
7	`COMMENT ON COLUMN`	`1050`
8	`CREATE SCHEMA`	`1054`
9	`DROP SCHEMA`	`1058`
10	`GRANT RDAREA`	`105C`
11	`GRANT SCHEMA`	`1060`
12	`GRANT ACCESS`	`1064`
13	`REVOKE RDAREA`	`1068`
14	`REVOKE SCHEMA`	`106C`
15	`REVOKE ACCESS`	`1070`
16	`CREATE VIEW`	`1074`
17	`DROP VIEW`	`1078`
18	`GRANT CONNECT`	`1098`
19	`GRANT DBA`	`109C`
20	`REVOKE CONNECT`	`10A0`
21	`REVOKE DBA`	`10A4`
22	`CREATE ALIAS`	`10A8`
23	`DROP ALIAS`	`10AC`
24	`CREATE PROCEDURE`	`10B0`
25	`DROP PROCEDURE`	`10B4`
26	`ALTER PROCEDURE`	`10B8`

No.	SQL statement	SQL type code (hexadecimal)
27	CREATE TYPE	10C0
28	DROP DATA TYPE	10C4
29	CREATE FUNCTION	10C8
30	DROP FUNCTION	10CC
31	ALTER ROUTINE	10E8
32	LOCK statement	2134
33	SELECT statement	4000, 4900, 4100, 4A04, 4204
34	INSERT statement	4108
35	UPDATE statement	4114
36	DELETE statement	411C
37	ASSIGN LIST statement	4224, 4324
38	DROP LIST statement	422C
39	PURGE TABLE statement	4184
40	CALL PROCEDURE	40BC
41	GET DIAGNOSTICS	40C0
42	Assignment statement	40C4
43	FREE LOCATOR statement	40C8

Table 14-12: Record format of a DAT-format file (global buffer pool statistical information)

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	—
2	Server name (`SERVER`)		8	
3	Global buffer name (`BUFFER NAME`)		16	
4	Statistics log acquisition time (`LOG GET TIME`)		14	Format: *MM*/*DD*/*hh*:*mm* when `-e sec` is omitted; *MM*/*DD*/*hh*:*mm*:*ss* when `-e sec` is specified
5	Number of buffer sectors in global buffer pool (`BUFFER COUNT`)	Numeric	10	—
6	Number of synchronization point dump pages (`SYNC POINT`)			
7	Maximum number of concurrently requested buffer sectors (`BUFFER MAX`)			
8	Update `GET`s count (`UPDATE GET`)			
9	Update buffer hits count (`UPDATE HIT`)			
10	Update buffer hit rate (`UPDATE HIT RATIO`)		3	Percentage
11	Update buffer flushes count (`UPDATE FLUSH`)		10	—
12	Reference `GET`s count (`REF GET`)			
13	Reference buffer hits count (`REF HIT`)			
14	Reference buffer hit rate (`REF HIT RATIO`)		3	Percentage

No.	Field name (title bar)	Attribute	Maximum length	Remarks
15	Reference buffer flushes count (`REF FLUSH`)		10	—
16	Real `READ`s count (`READ`)			
17	Input waits count (`WAIT READ`)			
18	Real `WRITE`s count (`WRITE`)			
19	Output waits count (`WAIT WRITE`)			
20	Buffer lock release waits count (`WAIT EXCLUSIVE`)			
21	Buffer shortages count (`BUFFER INSUFFICIENCY`)			
22	Prefetch input pages count (`PRRED`)			
23	Prefetch hit pages count (`PRHIT`)			
24	Prefetch hit ratio (`PRHIT RATIO`)		3	Percentage
25	Prefetch execution resources shortage count (`PR INSUFFICIENCY`)		10	—
26	Global buffer hit ratio (`BUFHIT RATIO`)		3	Percentage

No.	Field name (title bar)	Attribute	Maximum length	Remarks
27	Current reference buffers count (`CUR REF BUFNUM`)		10	—
28	Current update buffers count (`CUR UPDATE BUFNUM`)			
29	Deferred write trigger update buffers count (`TRG UPDATE BUFNUM`)			
30	Synchronization points count (`SYNC COUNT`)			
31	Number of prefetch `READ` requests (`PR READ REQ CNT`)			
32	Number of `READ` requests for LOB global buffer (`LOB READ REQ CNT`)			
33	Number of `WRITE` requests for LOB global buffer (`LOB WRITE REQ CNT`)			
34	Batch input pages count for LOB global buffer (`LOB BLK READ PGNUM`)			
35	Batch output pages count for LOB global buffer (`LOB BLK WRITE PGNUM`)			
36	System-specific information (`CACHE BUFFER INSUFFICIENCY`)			
37	System-specific information (`CACHE BUFFER FLUSH MAX`)			
38	System-specific information (`CACHE BUFFER FLUSH AVERAGE`)			
39	System-specific information (`LOCK WAIT CNT`)			Exponent representation
40	Percentage of lock contention-release wait (`LOCK WAIT RATIO`)		5	% (up to 1 decimal place)
41	System-specific information (`SLEEP EXEC AVERAGE`)		10	Exponent representation
42	Percentage rate of spin loop (`SPIN EXEC RATIO`)		5	% (up to 1 decimal place)
43	Average spin loop count (`SPIN EXEC AVERAGE`)		10	Exponent representation

No.	Field name (title bar)	Attribute	Maximum length	Remarks
44	Buffer pool lock exclusive time during synchronization point processing		10	—
45	No. 44 in microseconds		6	Value in seconds is not included
46	Number of buffers processed within buffer pool lock exclusive time during synchronization point processing		10	—
47	Take-over count of database write processing by reference request hit during synchronization point processing			
48	Take-over count of database write processing by update request hit during synchronization point processing			

Table 14-13: Record format of a DAT-format file (statistical information on HiRDB files for database manipulation)

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	—
2	Server name (`SERVER`)		8	
3	HiRDB file name (`FILE NAME`)		167	
4	RDAREA name (`RDAREA NAME`)		30	
5	Statistics log acquisition time (`LOG GET TIME`)		14	Format: *MM*/*DD*/*hh*:*mm* when `-e sec` is omitted; *MM*/*DD*/*hh*:*mm*:*ss* when `-e sec` is specified
6	Synchronous `READ`s count (`SYNC READ`)	Numeric	10	—
7	Synchronous `WRITE`s count (`SYNC WRITE`)			
8	System-specific information (`AIO READ`)			
9	Asynchronous `WRITE`s count (`AIO WRITE`)			
10	System-specific information (`LIST IO`)			
11	Opens count (`OPEN`)			
12	Closes count (`CLOSE`)			
13	I/O errors count (`IO ERROR`)			

Table 14-14: Record format of a DAT-format file (deferred write processing statistical information)

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	—
2	Server name (`SERVER`)		8	
3	Log acquisition time (`LOG GET TIME`)		11	Format: *MM/DD/hh:mm:ss*
4	Cause (`CAUSE`)		1	Trigger: `T` Pre-sync: `P` Synchronization point: `S` Database synchronization point: `D` RDAREA synchronization point: `R`
5	Pre-sync completion status (`STATUS`)*			Pre-sync completed: `P` Complete and terminated: `E`
6	I/O parallel level (`PARALLEL`)	Numeric	10	—
7	Total number of pages output (`OUT PAGE`)			Total number of pages output by deferred write processing for each cause
8	`WRITE` count (`DWEXEC`)			—
9	Minimum unit value of `WRITE` (`DWMIN`)			In seconds (rounded off)
10	No. 9 in microseconds (`DWMINM`)		6	Value in seconds is not included.
11	Maximum unit value of `WRITE` (`DWMAX`)		10	In seconds (rounded off)
12	No. 11 in microseconds (`DWMAXM`)		6	Value in seconds is not included.
13	Average unit value of `WRITE` (`DWAVG`)		10	In seconds (rounded off)

No.	Field name (title bar)	Attribute	Maximum length	Remarks
14	No. 13 in microseconds (DWAVGM)		6	Value in seconds is not included.
15	Total WRITE time (DWSUM)		10	In seconds (rounded off)
16	No. 15 in microseconds (DWSUMM)		6	Value in seconds is not included.
17	Parallel WRITE time (DWPARA)		10	In seconds (rounded off)
18	No. 17 in microseconds (DWPARAM)		6	Value in seconds is not included.
19	Execution time (DWTOTAL)		10	In seconds (rounded off)
20	No. 17 in microseconds (DWTOTALM)		6	Value in seconds is not included.
21	System-specific information (DWLSNGET)		10	In seconds (rounded off)
22	System-specific information (DWLSNGETM)		6	Value in seconds is not included.
23	System-specific information (DWSYNCLIST)		10	In seconds (rounded off)
24	System-specific information (DWSYNCLISTM)		6	Value in seconds is not included.
25	System-specific information (DWSEMOP)		10	In seconds (rounded off)
26	System-specific information (DWSEMOPM)		6	Value in seconds is not included.

Note

If the WRITE count is fewer than the total number of pages output (OUT PAGE), information is output in the batch mode. When information is output in the batch mode, the unit value of WRITE in Nos. 9 through 14 may be greater than when information is not output in the batch mode.

* This information is displayed when the cause is set to P (pre-sync); otherwise, the null value is displayed.

Table 14-15: Record format of a DAT-format file (SQL static optimization information)

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	--
2	UAP name (`UAP NAME`)			
3	Service name (`SERVICE NAME`)		31	
4	Front-end server name (`FES NAME`)		8	
5	Connection number (`CONNECT NO`)	Numeric	10	Serial number assigned to each connection in one front-end server
6	SQL counter (`SQL COUNTER`)			--
7	Front-end server process ID (`FES PROCESS ID`)			
8	Operation code (`OPERATION CODE`)	Character	4	For the operation codes, see *Table 14-10*.
9	Section number (`SECTION NO`)	Numeric	5	--
10	SQL type code (`SQL TYPE CODE`)	Character	4	For SQL type codes, see *Table 14-11*.
11	Server name (`SERVER NAME`)		8	--
12	Syntax analysis start time (`PARSE START(S)`)		14	Format: *MM/DD/hh:mm:ss*
13	No. 12 in microseconds (`PARSE START (MICRO)`)	Numeric	6	Value in seconds is not included.
14	Syntax analysis termination time (`PARSE END(S)`)	Character	14	Format: *MM/DD/hh:mm:ss*
15	No. 14 in microseconds (`PARSE END (MICRO)`)	Numeric	6	Value in seconds is not included.
16	Syntax analysis execution time (`PARSE EXEC TIME(SEC)`)		10	In seconds (value is rounded off)
17	No. 16 in microseconds (`PARSE EXEC TIME(MICRO)`)			Value in seconds is included.
18	Semantic analysis start time (`SEMAN START(S)`)	Character	14	Format: *MM/DD/hh:mm:ss*

No.	Field name (title bar)	Attribute	Maximum length	Remarks
19	No. 18 in microseconds (`SEMAN START(MICRO)`)	Numeric	6	Value in seconds is not included.
20	Semantic analysis termination time (`SEMAN END(S)`)	Character	14	Format: *MM/DD/hh:mm:ss*
21	No. 20 in microseconds (`SEMAN END(MICRO)`)	Numeric	6	Value in seconds is not included.
22	Semantic analysis execution time (`SEMAN EXEC TIME(SEC)`)		10	In seconds (Value is rounded off.)
23	No. 22 in microseconds (`SEMAN EXEC TIME(MICRO)`)			Value in seconds is included.
24	Optimization start time (`OPT START(S)`)	Character	14	Format: *MM/DD/hh:mm:ss*
25	No. 24 in microseconds (`OPT START(MICRO)`)	Numeric	6	Value in seconds is not included.
26	Optimization termination time (`OPT END(S)`)	Character	14	Format: *MM/DD/hh:mm:ss*
27	No. 26 in microseconds (`OPT END(MICRO)`)	Numeric	6	Value in seconds is not included.
28	Optimization execution time (`OPT EXEC TIME(SEC)`)		10	In seconds (value is rounded off)
29	No. 28 in microseconds (`OPT EXEC TIME(MICRO)`)			Value in seconds is included.
30	SQL object generation start time (`SQLOBJ GEN START(S)`)	Character	14	Format: *MM/DD/hh:mm:ss*
31	No. 30 in microseconds (`SQLOBJ GEN START(MICRO)`)	Numeric	6	Value in seconds is not included.
32	SQL object generation termination time (`SQLOBJ GEN END(S)`)	Character	14	Format: *MM/DD/hh:mm:ss*
33	No. 32 in microseconds (`SQLOBJ GEN END(MICRO)`)	Numeric	6	Value in seconds is not included.
34	SQL object generation execution time (`SQLOBJ GEN EXEC TIME(SEC)`)		10	In seconds (value is rounded off)
35	No. 34 in microseconds (`SQLOBJ GEN EXEC TIME(MICRO)`)			Value in seconds is included.

No.	Field name (title bar)	Attribute	Maximum length	Remarks
36	SQL object registration start time (`SQLOBJ ENT START(S)`)	Character	14	Format: *MM/DD/hh:mm:ss*
37	No. 36 in microseconds (`SQLOBJ ENT START(MICRO)`)	Numeric	6	Value in seconds is not included.
38	SQL object registration termination time (`SQLOBJ ENT END(S)`)	Character	14	Format: *MM/DD/hh:mm:ss*
39	No. 38 in microseconds (`SQLOBJ ENT END(MICRO)`)	Numeric	6	Value in seconds is not included.
40	SQL object registration execution time (`SQLOBJ ENT EXEC TIME(SEC)`)		10	In seconds (value is rounded off)
41	No. 40 in microseconds (`SQLOBJ ENT EXEC TIME(MICRO)`)			Value in seconds is included.

Table 14-16: Record format of a DAT-format file (SQL dynamic optimization information)

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	—
2	UAP name (`UAP NAME`)			
3	Service name (`SERVICE NAME`)		31	
4	Front-end server name (`FES NAME`)		8	
5	Connection number (`CONNECT NO`)	Numeric	10	Serial number assigned to each connection in one front-end server
6	SQL counter (`SQL COUNTER`)			—
7	Front-end server process ID (`FES PROCESS ID`)			
8	Operation code (`OPERATION CODE`)	Character	4	For operation codes, see *Table 14-10*.
9	Section number (`SECTION NO`)	Numeric	5	—
10	SQL type code (`SQL TYPE CODE`)	Character	4	For SQL type codes, see *Table 14-11*.
11	Server name (`SERVER NAME`)		8	—
12	Dynamic optimization start time (`DYNAMIC START`)		14	Format: *MM/DD/hh:mm:ss*
13	No. 12 in microseconds (`START(MICRO)`)	Numeric	6	Value in seconds is not included.
14	Dynamic optimization termination time (`DYNAMIC END`)	Character	14	Format: *MM/DD/hh:mm:ss*

No.	Field name (title bar)	Attribute	Maximum length	Remarks
15	No. 14 in microseconds (END(MICRO))	Numeric	6	Value in seconds is not included.
16	Dynamic optimization execution time (DYNAMIC EXEC TIME(SEC))		10	In seconds
17	No. 16 in microseconds (DYNAMIC EXEC TIME(MICRO))			Value in seconds is included.
18	System-specific information (SUB SECTION COUNT)		5	—
19	System-specific information (SUB SECTION NO)			
20	SQL object type (SQLOBJ KIND)	Character	1	F: Front end H: Foreign table access J: Sort/join T: Base table scan
21	Requested servers count (REQ SERVER)	Numeric	10	—
22	Allocated servers count (ALC SERVER)			
23	Allocated server name (ALC SERVER NAME)	Character	8	
24	RDAREA ID (RDAREA ID)	Numeric	10	

Table 14-17: Record format of a DAT-format file (SQL object execution information)

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	—
2	UAP name (`UAP NAME`)			
3	Service name (`SERVICE NAME`)		31	
4	Front-end server name (`FES NAME`)		8	
5	Connection number (`CONNECT NO`)	Numeric	10	Serial number assigned to each connection in one front-end server.
6	SQL counter (`SQL COUNTER`)			—
7	Front-end server process ID (`FES PROCESS ID`)			
8	Server name (`SERVER NAME`)	Character	8	
9	Thread number (`THREAD NO`)	Numeric	5	
10	Operation code (`OPERATION CODE`)	Character	4	For operation codes, see *Table 14-10*.
11	Section number (`SECTION NO`)	Numeric	5	—
12	System-specific information (`SUB SECTION NO`)			
13	SQL code (`SQLCODE`)		10	
14	SQL object execution start time (`SQLOBJ START`)	Character	14	Format: *MM/DD/hh:mm:ss*
15	No. 14 in microseconds (`START(MICRO)`)	Numeric	6	Value in seconds is not included.
16	SQL object execution termination time (`SQLOBJ END`)	Character	14	Format: *MM/DD/hh:mm:ss*
17	No. 16 in microseconds (`END(MICRO)`)	Numeric	6	Value in seconds is not included.
18	SQL object execution time (`SQLOBJ EXEC TIME(SEC)`)		10	In seconds (value is rounded off)
19	No. 18 in microseconds (`SQLOBJ EXEC TIME(MICRO)`)			Value in seconds is included.

No.	Field name (title bar)	Attribute	Maximum length	Remarks
20	Number of rows retrieved for SQL object (FETCH ROW)			--
21	Number of rows inserted for SQL object (INSERT ROW)			
22	Number of rows updated for SQL object (UPDATE ROW)			
23	Number of rows deleted for SQL object (DELETE ROW)			
24	Number of work table rows created (CREATE LIST)			
25	Number of work table rows deleted (DROP LIST)			
26	Send operations count (SEND)			
27	Length of send data (SEND DATA SIZE)			
28	Receive operations count (RECEIVE)			
29	Length of receive data (RECEIVE DATA SIZE)			
30	Number of times work tables were created (CREATE TMP LIST)			
31	Number of times work tables were deleted (DROP TMP LIST)			
32	HiRDB file output operations count (IOS OUTPUT)			
33	Buffer input operations count (BUFFER INPUT)			
34	SELECT APSL number (SEL APSL NO)			
35	SQL object type (SQLOBJ KIND)	Character	1	F: Front end J: Sort/join T: Base table scan

No.	Field name (title bar)	Attribute	Maximum length	Remarks
36	Number of times processing requests were issued to foreign server (`FOREIGN REQUEST COUNT`)	Numeric	10	--
37	Foreign server processing time (`FOREIGN EXEC TIME`)			In seconds
38	No. 37 in microseconds (`FOREIGN EXEC TIME(MICRO)`)			In microseconds (includes the value in seconds)
39	Number of rows retrieved on the foreign server (`FOREIGN FETCH ROW`)			--
40	Number of rows inserted in the foreign server (`FOREIGN INSERT ROW`)			--
41	Number of rows updated on the foreign server (`FOREIGN UPDATE ROW`)			--
42	Number of rows deleted on the foreign server (`FOREIGN DELETE ROW`)			--

Note

Nos. 36-42 are output when the HiRDB External Data Access facility is used. If this facility is not used, `0` or `****` is output.

Table 14-18: Record format of a DAT-format file (SQL object transfer statistical information)

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	—
2	UAP name (`UAP NAME`)			
3	Front-end server name (`FES NAME`)		8	
4	Connection number (`CONNECT NO`)	Numeric	10	
5	Front-end server process ID (`FES PROCESS ID`)			
6	Server name (`SERVER`)	Character	8	—
7	System-specific information (`SERVICE KIND`)		1	—
8	Operation code (`OPERATION CODE`)		4	For operation codes, see *Table 14-10*.
9	Section number (`SECTION NO`)	Numeric	5	—
10	System-specific information (`SUB SECTION NO`)			
11	SQL type code (`SQL TYPE CODE`)	Character	4	For SQL type codes, see *Table 14-11*.
12	SQL code (`SQLCODE`)	Numeric	10	—
13	Service start time (`SERVICE START`)	Character	14	Format: *MM/DD/hh:mm:ss*
14	No. 13 in microseconds (`SERVICE START(MICRO)`)	Numeric	6	Value in seconds is not included.
15	Service termination time (`SERVICE END`)	Character	14	Format: *MM/DD/hh:mm:ss*
16	No. 15 in microseconds (`SERVICE END(MICRO)`)	Numeric	6	Value in seconds is not included.
17	Service execution time (`SERVICE EXEC TIME(SEC)`)		10	In seconds (value is rounded off)
18	No. 17 in microseconds (`SERVICE EXEC TIME(MICRO)`)			Value in seconds is included.
19	SQL object transfer start time (`TRANSFER START`)	Character	14	Format: *MM/DD/hh:mm:ss*

No.	Field name (title bar)	Attribute	Maximum length	Remarks
20	No. 19 in microseconds (`TRANSFER START(MICRO)`)	Numeric	6	Value in seconds is not included.
21	SQL object transfer end time (`TRANSFER END`)	Character	14	Format: *MM/DD/hh:mm:ss*
22	No. 21 in microseconds (`TRANSFER END(MICRO)`)	Numeric	6	Value in seconds is not included.
23	SQL object transfer time (`TRANSFER TIME(SEC)`)		10	In seconds (value is rounded off)
24	No. 23 in microseconds (`TRANSFER TIME(MICRO)`)			Value in seconds is included.
25	SQL object transfer size (`TRANSFER SIZE`)		10	In bytes

Notes

1. If `SERVICE KIND` is `N`, `P`, or `R`, the utility sets a null value (`0` or `" "`) in items 8 to 11.
2. The utility sets a null value (`0` or `" "`) in items 19 to 25 in the following cases:
  - The SQL object was found in the SQL object buffer.
  - An error occurred before the SQL object is transferred (the SQL code is a non-zero value).

Table 14-19: Record format of a DAT-format file (SQL statement statistical information)

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	--
2	UAP name (`UAP NAME`)			
3	Module name (`MODULE NAME`)			
4	Service name (`SERVICE NAME`)		31	
5	Front-end server name (`FES NAME`)		8	
6	Connection number (`CONNECT NO`)	Numeric	10	Serial number assigned to each connection in one front-end server
7	SQL counter (`SQL COUNTER`)			--
8	Front-end server process ID (`FES PROCESS ID`)			
9	Operation code (`OPERATION CODE`)	Character	4	For operation codes, see *Table 14-10*.
10	Section number (`SECTION NO`)	Numeric	5	--
11	SQL length (`SQL LENGTH`)		10	
12	SQL statement (`SQL STRING`)	Character	2M	An SQL statement is output a maximum of 30 KB from the beginning.

Note

SQL statement history statistics are output only when the operation code is one of the following: `AUXI`, `OPN2`, `CALL`, `AUX2`, or `SET`.

Table 14-20: Record format of a DAT-format file (CONNECT/DISCONNECT statistical information)

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (HOST)	Character	32	—
2	UAP name (UAP NAME)			
3	Connect/disconnect time (CONNECT/DISCONNECT TIME)		14	Format: *MM/DD/ hh:mm:ss*
4	No. 3 in microseconds (CONNECT/DISCONNECT TIME(MICRO))	Numeric	6	Value in seconds is not included.
5	Front-end server process ID (FES PROCESS ID)		10	—
6	SQL code (SQLCODE)			
7	Connect/disconnect status (CONNECT/ DISCONNECT STATUS)	Character	1	C: CONNECT D: DISCONNECT
8	Client group ID (CLT GROUP)			1: WS client 2: PC client 3: Mainframe-based client 8: X/Open XA interface 9: Distributed client 0: Utility #: Other blank: None (space: 0×20)
9	Privilege information (PRIVILEGE)			C: CONNECT privilege D: DBA privilege blank: None (space: 0×20)
10	Connect/disconnect type (CONNECT/DISCONNECT KIND)			N: Normal (CONNECT/ DISCONNECT) S: set session authorization I: Internal disconnect T: Transaction resolution (OLTP only)

No.	Field name (title bar)	Attribute	Maximum length	Remarks
11	Authorization identifier (`USER NAME`)		30	—
12	Password (`PASSWORD`)			Upon `KFPA11560-E` error (invalid password)
13	CLT IP address (`CLT IP ADDRESS`)		15	Format: *XXX.XXX.XXX.XXX*

Table 14-21: Foreign server operation statistical information

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	--
2	Server name (`SERVER NAME`)		8	
3	Foreign server ID (`FOREIGN SERVER ID`)	Numeric	10	--
4	Foreign server name (`FOREIGN SERVER`)	Character	30	
5	DBMS type of the foreign server (`DBMS KIND`)		1	P: HiRDB R: XDM/RD O: Oracle I: DB2

No.	Field name (title bar)	Attribute	Maximum length	Remarks
6	Number of times connection was established (CONNECT COUNT)	Numeric	10	—
7	Number of connection errors (CONNECT ERROR COUNT)			
8	Number of connection errors due to number of licenses exceeded (LICENSE OVER COUNT)			
9	Commits count (COMMIT COUNT)			
10	Rollbacks count (ROLLBACK COUNT)			
11	Number of rows retrieved (FETCH ROW)			
12	Information used by the system (DBA CONNECT COUNT)			
13	Information used by the system (DBA DISCONNECT COUNT)			
14	Information used by the system (DBA PREPARE COUNT)			
15	Information used by the system (DBA OPEN COUNT)			
16	Information used by the system (DBA FETCH COUNT)			
17	Information used by the system (DBA CLOSE COUNT)			
18	Information used by the system (DBA EXECUTEIMD COUNT)			
19	Information used by the system (DBA EXECUTE COUNT)			
20	Information used by the system (DBA LOCKTABLE COUNT)			
21	Foreign server processing time (FOREIGN EXEC TIME)			In seconds
22	No. 21 in microseconds (FOREIGN EXEC TIME(MICRO))			In microseconds (includes the value in seconds)
23	Foreign server processing request count (FOREIGN EXEC COUNT)			—

No.	Field name (title bar)	Attribute	Maximum length	Remarks
24	Number of rows inserted in the foreign server (`FOREIGN INSERT ROW`)			—
25	Number of rows updated on the foreign server (`FOREIGN UPDATE ROW`)			—
26	Number of rows deleted on the foreign server (`FOREIGN DELETE ROW`)			—

Table 14-22: Foreign server utilization statistical information

No.	Field name (title bar)	Attribute	Maximum length	Remarks
1	Host name (`HOST`)	Character	32	--
2	UAP name (`UAP NAME`)		30	
3	Service name (`SERVICE NAME`)		31	
4	Front-end server name (`FES NAME`)		8	
5	Connection number (`CONNECT NO`)	Numeric	10	--
6	SQL counter (`SQL COUNTER`)			
7	Front-end server's process ID (`FES PROCESS ID`)			
8	Back-end server name (`BES NAME`)	Character	8	--
9	Back-end server's process ID (`BES PROCESS ID`)	Numeric	10	--
10	Thread number (`THREAD ID`)			
11	Section number (`SECTION NO`)		5	
12	Information used by the system (`SUB SECTION NO`)			
13	Foreign server ID (`FOREIGN SERVER ID`)		10	
14	Foreign server name (`FOREIGN SERVER`)	Character	30	--
15	SQL type code (`SQL TYPE CODE`)[1]		4	
16	Array `FETCH` count on the foreign server (`ARRAY FETCH ROW`)	Numeric	10	--
17	`FETCH` count on the foreign server (`ARRAY FETCH COUNT`)			
18	Number of rows retrieved on the foreign server (`FETCH ROW`)			
19	SQL execution start time at the foreign server (`EXEC START`)	Character	14	Format: *MM*/*DD*/*hh*:*mm*:*ss*
20	No. 19 in microseconds (`START(MICRO)`)	Numeric	6	Value in seconds is not included.
21	SQL execution end time at the foreign server (`EXEC END`)	Character	14	Format: *MM*/*DD*/*hh*:*mm*:*ss*

No.	Field name (title bar)	Attribute	Maximum length	Remarks
22	No. 21 in microseconds (`END(MICRO)`)	Numeric	6	Value in seconds is not included.
23	SQL execution time at the foreign server (`EXEC TIME(SEC)`)		10	In seconds
24	No. 23 in microseconds (`EXEC TIME(MICRO)`)			In microseconds (includes the value in seconds)
25	Foreign server processing request count (`EXEC COUNT`)			--
26	Number of rows inserted in the foreign server (`FOREIGN INSERT ROW`)			--
27	Number of rows updated on the foreign server (`FOREIGN UPDATE ROW`)			--
28	Number of rows deleted on the foreign server (`FOREIGN DELETE ROW`)			--

[1] For details about `SQL TYPE CODE`, see *Table 14-11*.

14.5 Notes

1. For the `pdstedit` command, return code `0` indicates normal termination, and return code `1` indicates abnormal termination.
2. If you selected `utf-8` as the character encoding in the `pdsetup` command, you may be able to use a file with a BOM as the input file for `pdstedit`. Table 14-23 shows whether or not files with a BOM can be used with `pdstedit`. Note that even when a file with a BOM is used as the input file for `pdstedit`, theBOM is skipped.

Table 14-23: Whether or not files with a BOM can be used in pdstedit (applicable to UTF-8)

Option	Input file	Use of file with a BOM
`-l`	Input statistics unload file	N
`-d`	Control information file	Y

Legend:

Y: Can be used

N: Cannot be used

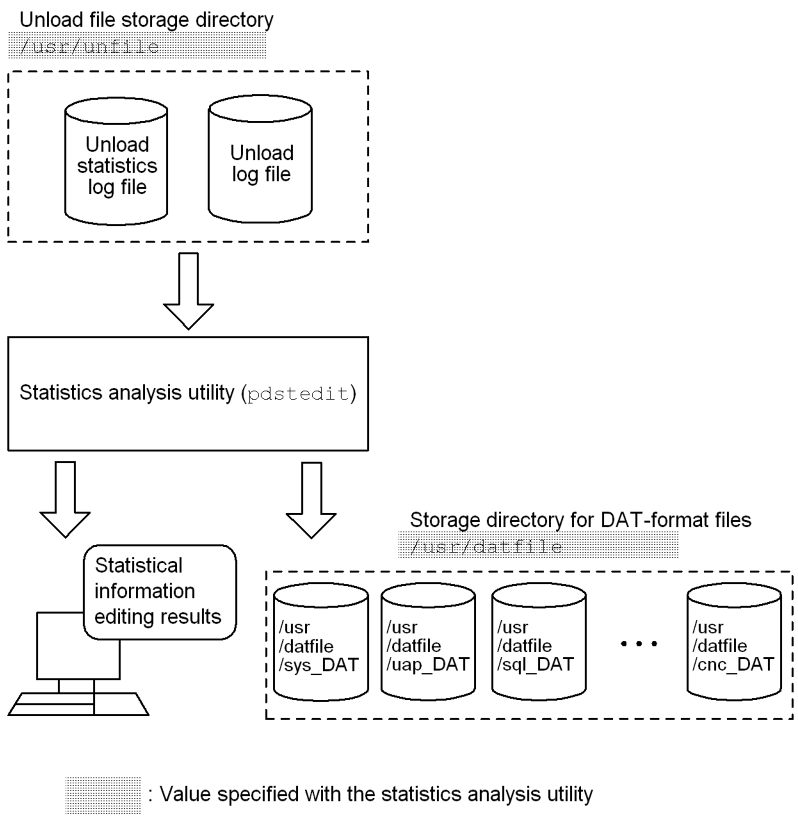
14.6 Examples

This section presents examples of using the statistics analysis utility.

Example 1

This example reads information in input statistics unload files and edits all statistical information. It outputs the information to a DAT-format file.

Overview


Unload file storage directory
/usr/unfile
Unload statistics log file
Unload log file
Statistics analysis utility (pdstedit)
Statistical information editing results
Storage directory for DAT-format files
/usr/datfile
/usr /datfile /sys_DAT
/usr /datfile /uap_DAT
/usr /datfile /sql_DAT
/usr /datfile /cnc_DAT
▨ : Value specified with the statistics analysis utility

Command execution

```
pdstedit -k all    ......................1
         -i /usr/unfile   ..............2
         -o /usr/datfile   .............3
```

Explanation

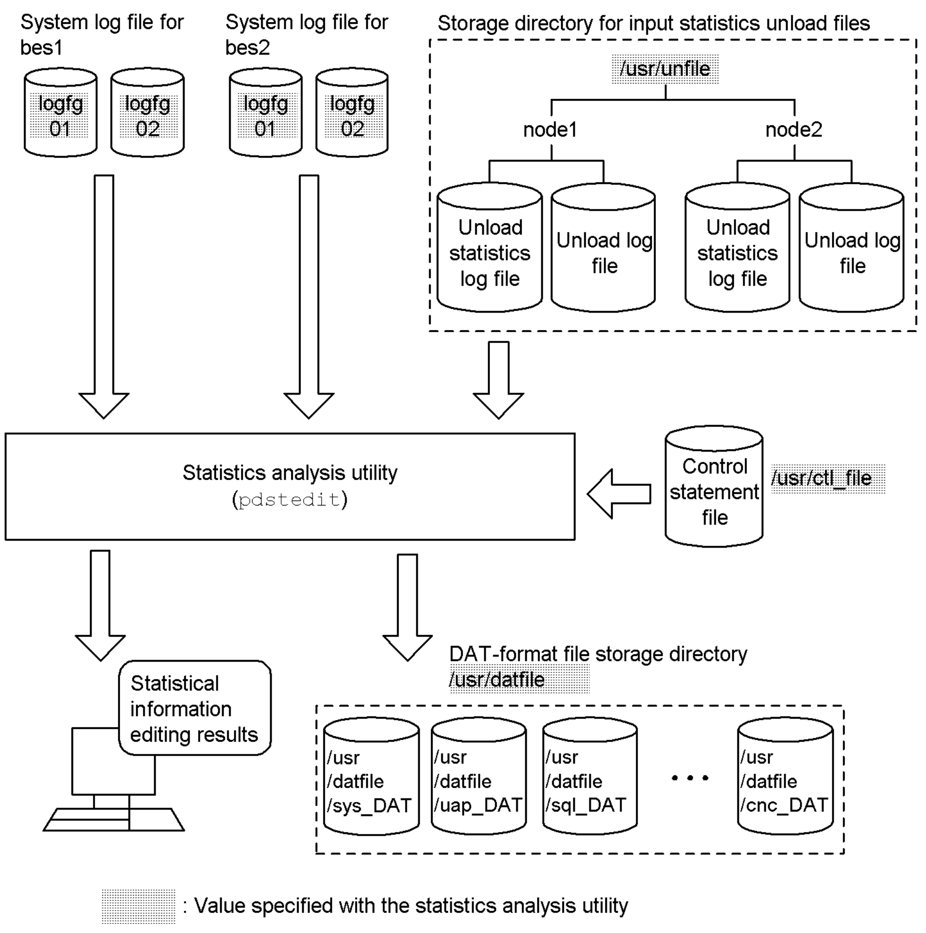
1. Specifies that all statistical information is to be edited.
2. Name of the directory containing the input statistics unload files: `/usr/unfile`
3. Name of the DAT-format file storage directory for storing the statistical information: `/usr/datfile`

Example 2

This example inputs the files listed below that can be referenced from the host executing `pdstedit` and then edits the statistical information. It also outputs the information to a DAT-format file.

- Multiple unload statistics log files and unload log files placed under a single directory
- System log files that have not been unloaded

Overview


System log file for bes1
logfg 01
logfg 02
System log file for bes2
logfg 01
logfg 02
Storage directory for input statistics unload files
/usr/unfile
node1
node2
Unload statistics log file
Unload log file
Unload statistics log file
Unload log file
Statistics analysis utility (pdstedit)
Control statement file
/usr/ctl_file
Statistical information editing results
DAT-format file storage directory
/usr/datfile
/usr /datfile /sys_DAT
/usr /datfile /uap_DAT
/usr /datfile /sql_DAT
/usr /datfile /cnc_DAT
: Value specified with the statistics analysis utility

Command execution

```
pdstedit -k all    ..........................1
         -i /usr/unfile   ..................2
         -d /usr/ctl_file   ................3
         -o /usr/datfile   .................4
```

Explanation

1. Specifies that all statistical information is to be edited.
2. Name of the directory containing the input statistics unload files: `/usr/unfile`

3. Name of the control statement file: `/usr/ctl_file`
4. Name of the DAT-format file storage directory for storing the statistical information: `/usr/datfile`

Contents of the control statement file (`/usr/ctl_file`)

```
file_group bes1:logfg01,logfg02
file_group bes2:logfg01,logfg02
```

Explanation

`bes1` and `bes2`:

Names of the servers corresponding to the specified file groups

`logfg01` and `logfg02`:

Names of the file groups that contain the system log files to be analyzed

Chapter

15. Database Condition Analysis Utility (pddbst)

This chapter explains the database condition analysis utility (`pddbst`), which analyzes the condition of the database, accumulates condition analysis results, and predicts when reorganization should be performed.

15.1 Overview

Executor: User with DBA privilege

`pddbst` provides the following three facilities:

- Database condition analysis facility
- Condition analysis result accumulation facility
- Facility for predicting reorganization time

15.1.1 Database condition analysis facility

The *database condition analysis facility* references the HiRDB directory, analyzes the storage status of the data dictionary tables or the tables and indexes in user RDAREAs, then displays the analysis results.

You can execute the database condition analysis facility in the following units:

- Condition analysis by RDAREA
- Condition analysis by table or index
- Storage condition analysis on cluster keys and clustering data pages by RDAREA

(1) Condition analysis by RDAREA

Condition analysis by RDAREA can be a logical analysis or a physical analysis.

Figures 15-1 (logical analysis) and 15-2 (physical analysis) show condition analysis by RDAREA.

Figure 15-1: Condition analysis by RDAREA: logical analysis

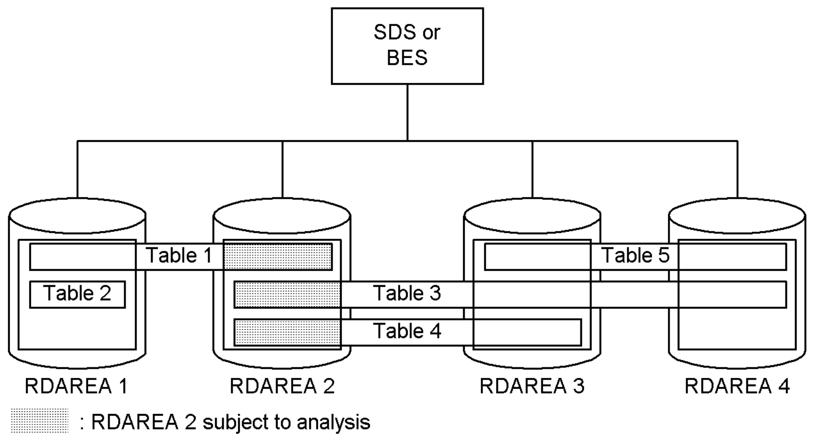
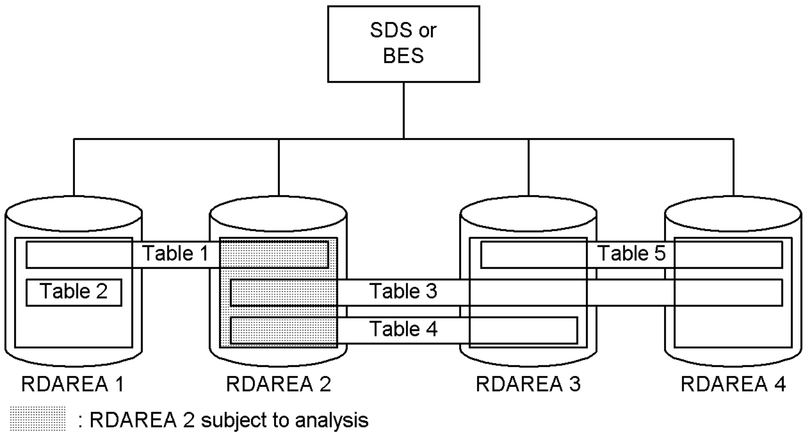


Figure 15-2: Condition analysis by RDAREA: physical analysis



(2) Condition analysis by table or index

Figure 15-3 shows condition analysis by table, and Figure 15-4 shows condition analysis by index.

Figure 15-3: Condition analysis by table

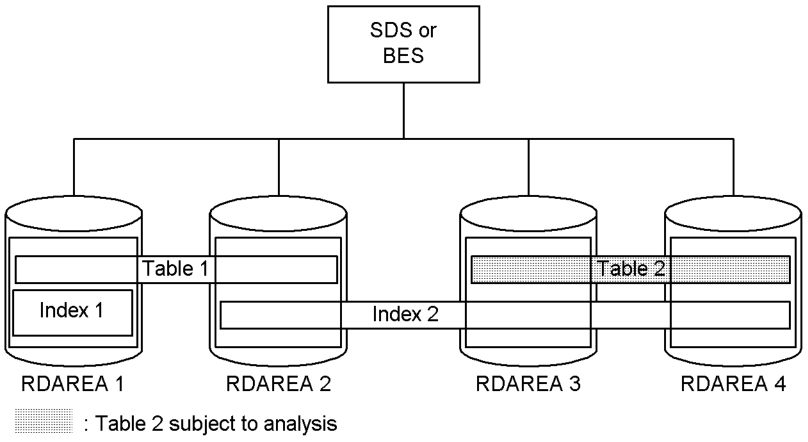
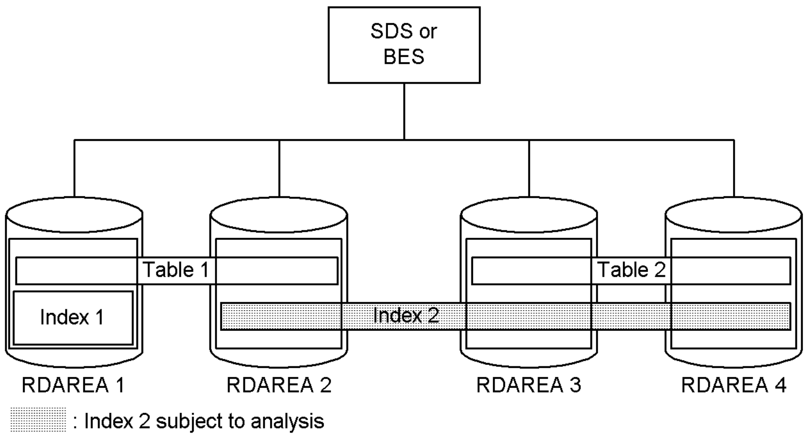


Figure 15-4: Condition analysis by index



(3) Storage condition analysis on cluster keys and clustering data pages by RDAREA

Cluster keys are analyzed and the incorrect-storage rate is displayed (the incorrect-storage rate indicates the degree of disorganization of tables or indexes).

The analysis results are displayed in the order of cluster keys and clustering data pages for each storage RDAREA.

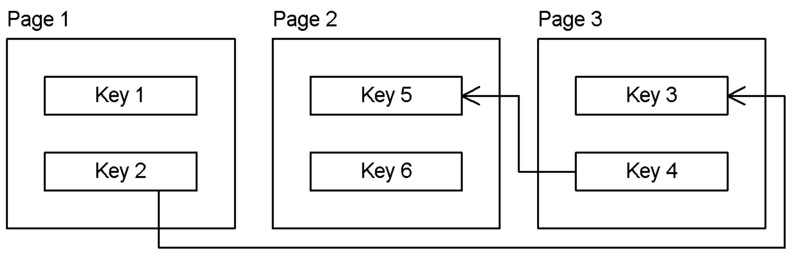
(a) Analysis by cluster key order

Cluster keys are retrieved in the order of the key values, and the analysis results are displayed as follows:

- Number of times the storage position changed (when storage positions span pages or segments)
- Number of times the storage order was incorrect (i.e., pages (segments) were stored in reverse order) against the number of times the storage position was changed; displayed in percentage (%) per page or per segment. The rate of incorrect storage order increases due to occurrences of page partitioning

Figure 15-5 shows the ratio of the number of storage position changes to the number of times the storage order was incorrect.

Figure 15-5: Ratio of number of storage position changes to number of times storage order was incorrect (%)



→ : Storage position change.

The storage position change from Key 4 to Key 5 is in reverse order, creating an incorrect storage order. The incorrect storage order rate for this figure is:

Number of times storage order is incorrect/number of storage position changes = 1/2 = 50%

(b) Analysis by clustering data page order

Clustering data pages are data pages in tables assigned to cluster keys. The storage position changes and the number of times the storage order was incorrect for row data when data in cluster key order was retrieved based on the storage position information for row data in the cluster key are displayed in the format "page unit/segment unit" (%).

When the storage condition becomes disorderly due to addition or updating of rows, either the incorrect storage order rate or the number of storage position changes, or both, will increase.

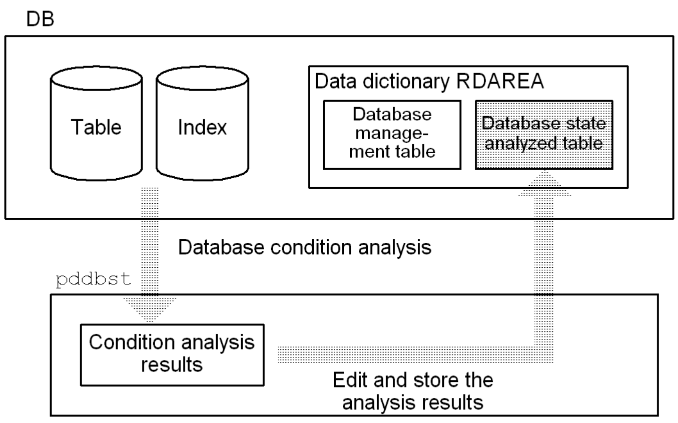
15.1.2 Condition analysis result accumulation facility

The *condition analysis result accumulation facility* accumulates database condition analysis results that are required for using the facility for predicting reorganization time.

The condition analysis result accumulation facility executes a logical analysis of the database condition by RDAREA, edits the analysis results, and then stores the results in the database state analyzed table. It is recommended that you execute the condition analysis result accumulation facility on a daily basis.

Figure 15-6 provides an overview of the condition analysis result accumulation facility.

Figure 15-6: Overview of condition analysis result accumulation facility


DB
Table
Index
Data dictionary RDAREA
Database manage- ment table
Database state analyzed table
Database condition analysis
pddbst
Condition analysis results
Edit and store the analysis results

There are two prediction levels in the condition analysis result accumulation facility, *prediction level 1* and *prediction level 2*. These levels correspond to prediction levels 1 and 2 for the facility for predicting reorganization time. Execution time is longer in prediction level 2 than in prediction level 1 because the former involves more detailed prediction. To execute the condition analysis result accumulation facility in prediction level 2, see *15.6 Executing the condition analysis result accumulation facility in prediction level 2*.

Prerequisites for using the condition analysis result accumulation facility:

To use the condition analysis result accumulation facility, you must make the following preparations:

1. Set `pd_rorg_predict=Y` in the system definition.
2. Use `pdmod` to create a data dictionary RDAREA for storing a database state analyzed table and a database management table.

15.1.3 Facility for predicting reorganization time

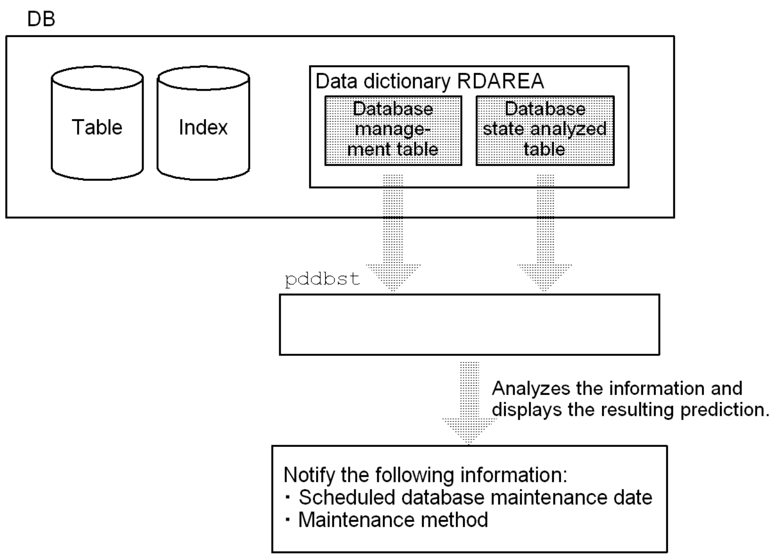
(1) About the facility for predicting reorganization time

The *facility for predicting reorganization time* analyzes information accumulated by the condition analysis result accumulation facility (in a database state analyzed table) and information in database management tables, predicts when RDAREA maintenance will be needed (scheduled database maintenance date), and recommends a maintenance date and method.

For details about the facility for predicting reorganization time, see the *HiRDB Version 8 System Operation Guide*.

Figure 15-7 provides an overview of the facility for predicting reorganization time.

Figure 15-7: Overview of the facility for predicting reorganization time


DB
Table
Index
Data dictionary RDAREA
Database management table
Database state analyzed table
pddbst
Analyzes the information and displays the resulting prediction.
Notify the following information:
・Scheduled database maintenance date
・Maintenance method

Note

If the RDAREA contains only one resource (for example, an RDAREA contains only a single table), the utility uses the database state analyzed table and database management table to make its prediction. If an RDAREA contains multiple resources, the utility uses only the database state analyzed table to make its prediction.

The facility for predicting reorganization time consists of the following two levels:

- Prediction level 1

  Predicts a shortage of RDAREA capacity.

- Prediction level 2

  Predicts deterioration of table and index storage efficiency as well as a shortage of RDAREA capacity.

(2) Prerequisites for using the facility for predicting reorganization time

To use the facility for predicting reorganization time, you must complete the following preparations:

1. Set `pd_rorg_predict=Y` in the system definition.
2. Use `pdmod` to create a data dictionary RDAREA for storing a database state analyzed table and a database management table.
3. Accumulate at least four sets of analysis information with the condition analysis

result accumulation facility.

(3) Predicting the reorganization time

The following describes how to predict the reorganization time.

(a) In prediction level 1

In prediction level 1, the facility predicts the date on which the percentage of used segments in an RDAREA is likely to exceed a standard value (this date is called the *scheduled database maintenance date*). The facility determines that maintenance will be required for an RDAREA when the standard value is predicted to be reached within a specified period (the monitoring interval[1]). When maintenance is determined to be required, the facility also recommends a maintenance method. The facility selects for the maintenance method either reorganization based on the specified maintenance extension period[1] or RDAREA expansion, whichever is best suited to the RDAREA.

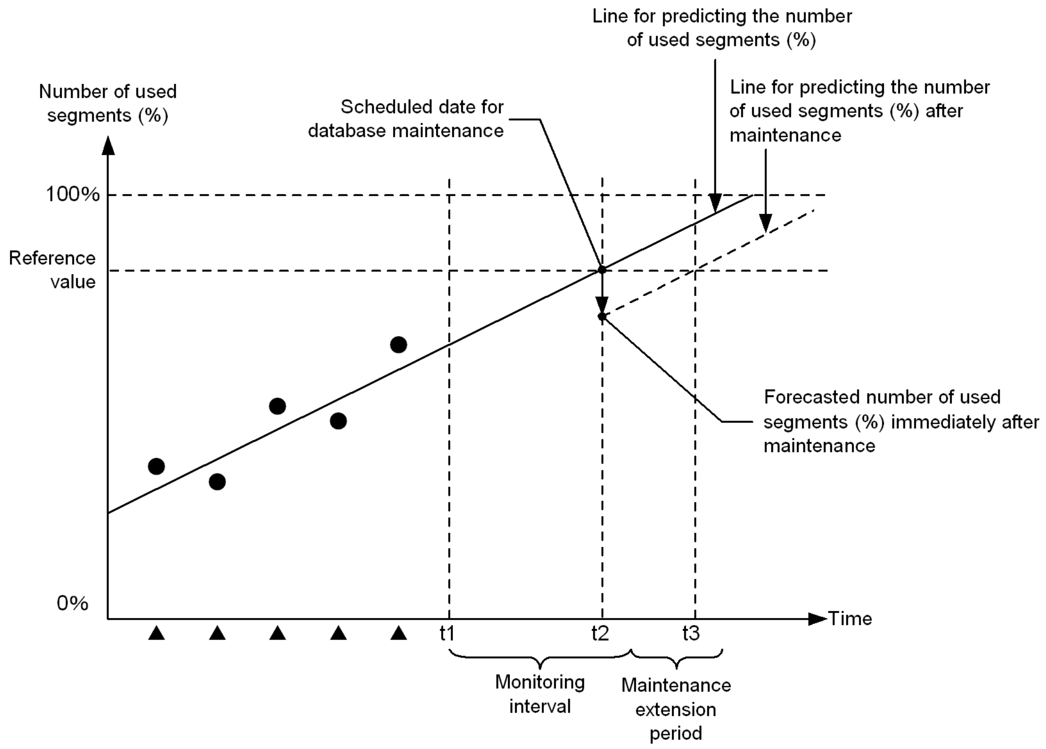
You can change the standard values by specifying the standard value definition file.[2]

[1] You specify a monitoring interval and maintenance extension period with the `-R` option.

[2] You use the `-c` option to specify the standard value definition file.

Figure 15-8 shows the relationships among the percentage of used segments in an RDAREA, the scheduled database maintenance day, the monitoring interval, and the maintenance extension period.

Figure 15-8: Relationships among the percentage of used segments in an RDAREA, the scheduled database maintenance date, the monitoring interval, and the maintenance extension period



Legend:

▲: Execution of the condition analysis result accumulation facility

●: Number of used segments (%), as determined by execution of the condition analysis result accumulation facility

`t1`: Execution of the facility for predicting reorganization time

`t2`: Date the prediction line for the number of used segments (%) is forecast to reach the standard value

`t3`: Date the prediction line for the number of used segments (%) is forecast to reach the standard value after maintenance has been performed

(b) In prediction level 2

In prediction level 2, the facility predicts any deterioration of table and index storage efficiency as well as any shortage of RDAREA capacity.

(4) Prediction of reorganization time when the database was not restored to its most recent status after a failure

If the database was not restored to its most recent status after a failure (such as when the database was restored to the point at which a backup was made or when only a specified range was restored), executing the facility for predicting reorganization time cannot result in a valid prediction. To use the facility for predicting reorganization time under such circumstances, you must *reset the accumulated condition analysis results*.

Suppose that a failure occurred subsequent to execution of the condition analysis result accumulation facility, and the database was restored to the point at which a backup was made (which was earlier than when the condition analysis result accumulation facility was executed). If the facility for predicting reorganization time is executed in this status, a valid prediction cannot be obtained because the contents of the database are older than the accumulated condition analysis results. In such a case, you need to reset the accumulated condition analysis results once you have restored the database to the point at which the backup was make. This will make the accumulated condition analysis results match the database contents, thereby enabling a valid prediction.

You specify the `-I` option to reset the condition analysis result accumulation facility.

15.2 Command format

15.2.1 Format

(1) For the database condition analysis facility

Condition analysis by RDAREA

```
pddbst -r RDAREA-name[,RDAREA-name]...
       [-k {logi [-d]|phys [-f] [-a [-h]]}] [-u user-ID [-p password]]
       [-q generation-number] [-b]
       [-X response-monitoring-time-for-server-to-server-communication]
       [-v control-statements-file-name]
```

Condition analysis by table

```
pddbst -t [authorization-identifier.]table-identifier
       [-s] [-u user-ID [-p password]] [-d]
       [-q generation-number] [-b]
       [-X response-monitoring-time-for-server-to-server-communication]
       [-v control-statements-file-name]
```

Condition analysis by index

```
pddbst -i [authorization-identifier.]index-identifier
       [-u user-ID [-p password]] [-d]
       [-q generation-number] [-b]
       [-X response-monitoring-time-for-server-to-server-communication]
       [-v control-statements-file-name]
```

Storage condition analysis on cluster keys and clustering data pages

```
pddbst -t [authorization-identifier.]table-identifier
       [-u user-ID [-p password]] -k clus
       [-q generation-number] [-b]
       [-X response-monitoring-time-for-server-to-server-communication]
       [-v control-statements-file-name]
```

(2) For the condition analysis result accumulation facility

```
pddbst -r {RDAREA-name|ALL} [-k logi] -e prediction-level
       [-w pause-time,segments-count] [-n analysis-segments-count] [-I]
       [-u authorization-identifier [-p password]]
       [-X response-monitoring-time-for-server-to-server-communication]
       [-v control-statements-file-name]
```

(3) For the facility for predicting reorganization time

```
pddbst [-r ALL] -k pred -e prediction-level
       [-m] [-R monitoring-interval[,maintenance-extension-period]]
       [-c standard-value-definition-file-name] [-u authorization-identifier [-p password]]
       [-X response-monitoring-time-for-server-to-server-communication]
       [-v control-statements-file-name]
```

15.2.2 Options

(1) `-r` *{RDAREA-name[,RDAREA-name]...|ALL}...*

~ <identifier> ((1-30))

Specifies the names of the RDAREAs to be analyzed. `ALL` can be specified only for the condition analysis result accumulation facility and the facility for predicting reorganization time.

You can specify the following RDAREA types:

- Data dictionary RDAREA
- Data dictionary LOB RDAREA
- User RDAREA
- User LOB RDAREA
- Registry RDAREA
- Registry LOB RDAREA

Rules

1. The same RDAREA name cannot be specified more than once.
2. If an RDAREA name is enclosed in double quotation marks (`"`), the command treats it as being case sensitive. If it is not enclosed in double quotation marks (`"`), the command treats it as in all uppercase letters.
3. If an RDAREA name contains a space, enclose the RDAREA name in double quotation marks (`"`). If you are using `sh` (Bourne shell), `csh` (C

shell), and `ksh` (Korn shell), you need to enclose the entire RDAREA name in single quotation marks (`'`).

4. You can specify a maximum of 16 RDAREA names. In the case of a logical analysis (`logi` specified in the `-k` option), specify only one RDAREA name (if multiple RDAREA names are specified, the utility uses only the first name specified).
5. If you are also specifying the `-q` option, specify the original RDAREA name.
6. The condition analysis result accumulation facility and the facility for predicting reorganization time use a single transaction to analyze all tables stored in an RDAREA. Therefore, the value of the `pd_max_access_tables` operand in the system definition must at least equal the number of tables in the RDAREA that contains the greatest number of tables among all target RDAREAs.

(2) *-k {logi|phys|clus|pred}*

Specifies the type of analysis that is to be executed (by the database condition analysis facility, condition analysis result accumulation facility, or facility for predicting reorganization time).

`logi`:

Specifies that RDAREA condition analysis (logical analysis) or the condition analysis result accumulation facility is to be executed.

`phys`:

Specifies that RDAREA condition analysis (physical analysis) is to be executed.

`clus`:

Specifies that storage condition analysis is to be executed on cluster keys and clustering data pages.

`pred`:

Specifies that the facility for predicting reorganization time is to be executed.

Combination of -k option and other options:

The values of the `-r`, `-t`, and `-i` options depend on the value of the `-k` option:

-k option value	Whether or not operation can be specified			Remarks
	-r	-t	-i	
logi	M	N	N	Condition analysis by RDAREA (logical analysis)
phys	M	N	N	Condition analysis by RDAREA (physical analysis)

-k option value	Whether or not operation can be specified			Remarks
	-r	-t	-i	
clus	N	M	N	Storage condition analysis on cluster keys and clustering data pages
Omitted	N	M	N	Condition analysis by table
	N	N	M	Condition analysis by index
	M	N	N	Condition analysis by RDAREA (logical analysis)
`pred`	O	N	N	Facility for predicting reorganization time

Legend:

M: Specification is mandatory

O: Specification is optional

N: Specification is not permitted

(3) `-d`

Specifies that detailed page information is to be displayed. This option is applicable to logical analysis (`-k logi`) and condition analysis by table and by index. Whether or not `pdrorg` and `pdreclaim` need to be executed depends on this detailed page information.

(4) `-f`

Specifies that HiRDB file information is to be displayed. This option is applicable to physical analysis (`-k phys`) only.

(5) `-a`

Specifies that the analysis results are to be displayed in DAT format.

When this option is specified, the utility assumes that the `-f` option is specified.

(6) `-h`

When the `-a` option is specified, specifies that a header is to be displayed.

(7) -u authorization-identifier

Specifies the authorization identifier of the user executing `pddbst`.

Rules

1. If this option is omitted, the authorization identifier and password defined in `PDUSER` in the client environment definition are assumed. If the `PDUSER`

value is not set, the user name corresponding to the OS user ID of the user executing the utility is assumed.

2. If a user ID is enclosed in double quotation marks (`"`), the command treats it as being case sensitive. If it is not enclosed in double quotation marks (`"`), the command treats it as in all uppercase letters. If you are using `sh` (Bourne shell), `csh` (C shell), or `ksh` (Korn shell), you must enclose the entire user ID in single quotation marks (`'`).

(8) `-p` *password*

Specifies the password for the authorization identifier specified in the `-u` option.

Rules

1. If this option is not specified and `PDUSER` is also not set in the client environment definition, the value entered on the utility execution screen is assumed.
2. If a password is enclosed in double quotation marks (`"`), the command treats it as being case sensitive. If it is not enclosed in double quotation marks (`"`), the command treats it as in all uppercase letters. If you are using `sh` (Bourne shell), `csh` (C shell), or `ksh` (Korn shell), you must enclose the entire password in single quotation marks (`'`).

(9) `-t` *[authorization-identifier.]table-identifier*

Specifies the name of the table to be analyzed.

Rules

1. This cannot be a view table or foreign table.
2. If the authorization identifier is omitted, the authorization identifier specified in the `-u` option is assumed.
3. If an authorization identifier or table identifier is enclosed in double quotation marks (`"`), the command treats it as being case sensitive. If it is not enclosed in double quotation marks (`"`), the command treats it as in all uppercase letters.
4. If a table identifier contains a space, enclose it in double quotation marks (`"`). If you are using `sh` (Bourne shell), `csh` (C shell), or `ksh` (Korn shell), you must enclose the entire identifier in single quotation marks (`'`).

(10) `-s`

Specifies that the display in the case of table condition analysis is to include the total number of lines stored in the table or the number of lines stored for each RDAREA.

(11) `-i` *[authorization-identifier.]index-identifier*

Specifies the name of the index to be analyzed.

Rules

1. This cannot be an external index.
2. If the authorization identifier is omitted, the authorization identifier specified in the `-u` option is assumed.
3. If an authorization identifier or index identifier is enclosed in double quotation marks (`"`), the command treats it as being case sensitive. If it is not enclosed in double quotation marks (`"`), the command treats it as in all uppercase letters.
4. If an index identifier contains a space, enclose it in double quotation marks (`"`). If you are using `sh` (Bourne shell), `csh` (C shell), and `ksh` (Korn shell), you need to enclose the entire identifier in single quotation marks (`'`).

(12) `-q` *generation-number*

~ <unsigned integer> ((0-10))

Specifies that information about the inner replica facility is to be displayed.

Rules

1. An error results if this option is specified when HiRDB Staticizer Option has not been installed.
2. When this option is omitted, the utility assumes that the following RDAREA is to be analyzed:
   - For condition analysis by RDAREA: RDAREA specified in the `-r` option
   - For condition analysis by table or index or storage condition analysis on cluster key and clustering data page: Current RDAREA
3. This option cannot be specified when you execute the condition analysis result accumulation facility or the facility for predicting reorganization time. When the inner replica facility is used, the utility assumes `-q 0`.

(13) `-b`

Specifies that the online performance of applications is to be protected from any adverse effects of executing `pddbst`. When this option is specified together with the `-s` option, the analysis time may increase. The `-b` option is ignored when searching data dictionary tables for a table and indexes that are to be the targets of database condition analysis, and when searching the operation management table by the facility for predicting reorganization time.

Criteria

`pddbst` first reads a resource page to be analyzed into the global buffer. If a space shortage occurs in the global buffer at that time, online performance may be affected adversely because a page that had already been read by an application may be swept out of the global buffer. To avoid such adverse effects on performance, specify the `-b` option.

When the `-b` option is specified, a page read by `pddbst` is always treated as the oldest page, so that it will be the first page swept out of the global buffer in the event of a space shortage. When the `-b` option is not specified, the LRU management method is used.

Advantages

Even if a space shortage occurs in the global buffer, there will be no adverse effects on the online performance of applications. However, during execution of `pddbst`, re-reading of directory pages may occur in order to acquire the requisite number of rows.

(14) -e prediction-level

Specifies the prediction level to be used when the condition analysis result accumulation facility or the facility for predicting reorganization time is applied.

`1`

Specifies that the condition analysis result accumulation facility or the facility for predicting reorganization time is to be executed in prediction level 1.

`2`

Specifies that the condition analysis result accumulation facility or the facility for predicting reorganization time is to be executed in prediction level 2.

Rules

1. Set the same prediction level for both the condition analysis result accumulation facility and the facility for predicting reorganization time.
2. When the `-e` option is specified, the utility assumes that the `-b` option is also specified.

(15) -w pause-time,segments-count

Specifies that the condition analysis result accumulation facility in prediction level 2 is to be executed with the interval analysis method. For details about interval analysis, see *15.6.1 Interval analysis*. This method pauses processing for the specified amount of time after the specified number of segments have been analyzed.

pause-time ~ <unsigned integer> ((10-60000))

Specifies the pause time, in units of 10 milliseconds. If the specified value is not an even 10 milliseconds, the value is rounded up (for example, 15 is rounded up to 20).

segments-count ~ <unsigned integer> ((1-2147483647))

Specifies the number of segments to be analyzed between pauses.

Rules

1. The `-w` option is applicable to prediction level 2 (`-e 2`). Specifying this option in the case of another prediction level results in a control statement error.
2. In the case of a HiRDB/Parallel Server, the specified value applies to all servers.
3. If the global buffer contains no page to be analyzed, as many input/output operations as (specified number of segments × segment size) occur until the pause time is reached.

(16) -n analysis-segments-count

~ <unsigned integer> ((2-10))

Specifies that the condition analysis result accumulation facility in prediction level 2 is to be executed with the merge analysis method. For details about merge analysis, see *15.6.2 Merge analysis*.

This method divides a single execution of the condition analysis result accumulation facility into the specified number of analysis segments. Data equivalent to 1/*n*, where *n* is the specified number of analysis segments, is the target of each analysis. This means that it takes *n* executions to analyze all of the target data.

Rules

1. The `-n` option is applicable to prediction level 2 (`-e 2`). Specifying this option in the case of another prediction level results in a control statement error.
2. When you specify the `-n` option, you must execute `pddbst` the specified number of times. If you change the `-n` option value before the current series of analyses has been completed, the utility ignores the analysis information that has been accumulated so far and starts the analyses over from the beginning.
3. If maintenance is performed on the target table or index before the entire series of analyses has been completed, the utility starts the analyses over from the beginning.

(17) -l

Specifies that the accumulated condition analysis results are to be reset. For details about resetting the accumulated condition analysis results, see *15.1.3(4) Prediction of reorganization time when the database was not restored to its most recent status after a failure*.

Rules

1. When the -I option is specified, ALL cannot be specified in the -r option.
2. Specify the -I option only the first time you execute the condition analysis result accumulation facility after data recovery. If the -I option is specified subsequently, all the accumulated condition analysis results will be reset.

(18) -m

Specifies that information about both the scheduled database maintenance date and the maintenance method is to be displayed by the facility for predicting reorganization time. When this option is omitted, the utility displays information about the scheduled database maintenance date only.

(19) -R *monitoring-interval*[,*maintenance-extension-period*]

When the facility for predicting reorganization time is executed, specifies the target period and the maintenance extension period.

monitoring-interval ~ <unsigned integer> ((1-400)) <<14>>

Specifies the target monitoring interval in days.

For example, to monitor an RDAREA for the next two weeks for scheduling a database maintenance date, specify 14.

maintenance-extension-period ~ <unsigned integer> ((0-400)) <<monitoring period/2>>

Specifies the minimum number of days that must elapse between scheduled database maintenance dates.

For example, once database maintenance has been executed, if you wish to require at least one week to elapse before the next database maintenance, specify 7.

(20) -c *standard-value-definition-file-name*

~ <path name> ((up to 1023 bytes))

When the facility for predicting reorganization time is executed, the utility makes its predictions based on standard values predefined for various items. To change any of the standard values, this option specifies the absolute or relative path name of the standard value definition file.

You specify in the standard value definition file the new standard values for the items. For details about the standard value definition file, see *15.5 Standard value definition file (facility for predicting reorganization time)*.

(21) -X response-monitoring-time-for-server-to-server-communication

~ <unsigned integer> ((1-65535)) ((300))

If an error, such as a communication error, occurs at the server where the command was executed, the command may stop responding and the application may stop. To help you detect errors, `pddbst` enables you to monitor a response time during communication for dictionary manipulation that is performed by the command.

You set in the `-X` option a response monitoring time during dictionary manipulation (in seconds). If the execution time during dictionary manipulation exceeds the value set in the `-X` option, `pddbst` treats it as a dictionary access error and cancels processing with return code `8`.

Criteria

- If you want to detect an error in less time than 300 seconds in the event of a no-response from the server due to a communication error or unit down, specify a value less than 300 in the `-X` option.
- If the system switchover facility is used, the command may keep waiting for a response even though system switchover has been completed. In such a case, you can terminate the command immediately by reducing the monitoring time.
- The specified monitoring time may result in a timeout if a response from the dictionary is delayed and the utility's preprocessing is not completed within 300 seconds, which is the default value of the `-X` option. This can happen when many applications and utilities are executing concurrently. In such an environment, specify a value greater than 300 in the `-X` option.

(22) `-v` *control-statement-file-name*

~ <path name> ((up to 1023 bytes))

Specifies the absolute or relative path name of the file that contains the `pddbst` control statements.

This file must be located on the host where `pddbst` is executed. You can specify the following control statements:

- `predict` statement
- `workdir` statement

15.2.3 predict statement

The `predict` statement is used to output the execution results of the facility for predicting reorganization time to a file in CSV format.

Rules

1. You can specify the `predict` statement only when you execute the facility for predicting reorganization time.
2. Only one `predict` statement can be specified in the control statement file.

(1) Format

```
predict file=CSV-output-file-name
```

(2) Explanation

(a) CSV-output-file-name

～ <path name> ((up to 1023 bytes))

Specifies the absolute path name of the regular file to which the results of the facility for predicting reorganization time are to be output.

For details about the format of the output file, see *15.4.2 Output format in CSV format*.

Rules

1. For a HiRDB/Parallel Server, create the file on the host where the system manager is located.
2. The HiRDB administrator must have access privileges to the specified file.
3. If the file name contains a space, enclose the entire file name in double quotation marks (`"`).
4. If the specified file is not found, the utility creates the file; if the file already exists, the utility overwrites it.

15.2.4 workdir statement

The `workdir` statement specifies the directory to which work files used during execution of `pddbst` are to be output. `pddbst` uses work files for executing database condition analysis and for sorting data.

Rules

1. Only one `workdir` statement can be specified in the control statement file.
2. When the `workdir` statement is omitted, the utility creates work files in the `/tmp` directory on the host that contains the single server or where the system manager is located. These files are deleted when `pddbst` is terminated; however, they may remain in the system if `pddbst` terminates abnormally.

(1) Format

`workdir dir=`*directory-name-for-work-files*

(2) Explanation

(a) directory-name-for-work-files

～ <path name> ((up to 1000 bytes))

Specifies the absolute path name of the directory to which work files are to be output.

Rules

1. For a HiRDB/Parallel Server, specify a directory on the host where the system manager is located.
2. The HiRDB administrator must have access privileges to the specified directory.
3. If the directory name contains a space, enclose the entire directory name in double quotation marks (`"`).
4. If the specified directory does not contain a needed file, the utility creates the file; if the directory already contains a needed file, the utility overwrites the existing file.

15.3 Output format of the database condition analysis facility

15.3.1 Condition analysis by RDAREA (logical analysis)

This utility analyzes the storage condition of all segments and pages for the tables and indexes that are stored in an RDAREA. Note, however, that for tables and indexes that have been partitioned and stored among several RDAREAs, this utility analyzes only those segments stored in the specified RDAREAs.

(1) Purpose

You can detect a disordered table or index and obtain the capacity status of a single RDAREA.

(2) Analysis results

The following shows the results of condition analysis by RDAREA (logical analysis):

(a) For an RDAREA other than a LOB RDAREA

```
pddbst VV-RR(Object Option) ** RD Area Logical Analysis **  2003/03/31 18:33:38 [1]
RD Area Name   : user_rdarea_1 [2]
 Server        : bes1 [3]
 Total Segment :       150[4] Segment Size :         5 Pages [6]
 Unused Segment:        24[5] Page Size    :      4096 Bytes [7]
 Original RD Area Name : user_rdarea_1 [21]
 Generation Number :  0[22]  Replica RD Area Count :  0 [23]
 [24]     [25]               [26]               [27]
 History1 Hold Status :      Hold Code :     0  Hold Time :
 History2 Hold Status :      Hold Code :     0  Hold Time :

Table Name : TBL01 [8]
Auth Id    : user1 [9]
Status     :   [20]
Reference Pending Status : P [33]
Check     Pending Status : P [34]
 Search Mode : INS [28] Segment Reuse :          - segments [29]
 Reuse Search Failure :          0/         0 [30]
         Used(Full)       Used(      Full)        Sum
         [10]  [11]        [12]       [13]        [14]
 Segment 100%(  0%)        104(         0)        104
         [15]  [16]        [17]       [18]        [19]
 Page    100%(  0%)        518(         0)        520

--------------------------------------------------------------------------------
 Collect Prearranged Page :         55 [31]
 Used Page Ratio       Page(Ratio) [32]
            0% :         57(  11%)
        1- 10% :         19(   4%)
       11- 20% :         28(   6%)
       21- 30% :         70(  14%)
       31- 40% :         95(  19%)
       41- 50% :         71(  14%)
```

```
      51- 60% :         69(  14%)
      61- 70% :         58(  12%)
      71- 80% :         37(   8%)
      81- 90% :         14(   3%)
      91-100% :          2(   1%)
Total                  520


Table Name : TBL02
Auth Id    : user1
Status     :
 Search Mode : INS   Segment Reuse :          - segments
 Reuse Search Failure :          0/         0
         Used(Full)       Used(      Full)        Sum
 Segment 100%(  0%)         21(         0)         21
 Page     98%(  0%)        103(         0)        105

 Collect Prearranged Page :          9
 Used Page Ratio       Page(Ratio)
           0% :         11(  11%)
        1- 10% :          1(   1%)
       11- 20% :          6(   6%)
       21- 30% :          6(   6%)
       31- 40% :         12(  12%)
       41- 50% :         13(  13%)

       51- 60% :          5(   5%)
       61- 70% :          7(   7%)
       71- 80% :         27(  26%)
       81- 90% :         12(  12%)
       91-100% :          5(   5%)
 Total                  105


Index Name : IDX_TBL01_C1
Auth Id    : user1
Status     :
         Used(Full)       Used(      Full)        Sum
 Segment 100%(  0%)          1(         0)          1
 Page    100%(  0%)          5(         0)          5

 Collect Prearranged Page :          0
 Collect On Page :          0
 Used Page Ratio       Page(Ratio)
           0% :          0(   0%)
        1- 10% :          1(  20%)
       11- 20% :          0(   0%)
       21- 30% :          3(  60%)
       31- 40% :          0(   0%)
       41- 50% :          1(  20%)

       51- 60% :          0(   0%)
       61- 70% :          0(   0%)
       71- 80% :          0(   0%)
       81- 90% :          0(   0%)
       91-100% :          0(   0%)
 Total                    5
```

Explanation:

1. Date and time of completion of acquisition of the information required for this condition analysis, in the format *YYYY/MM/DD hh:mm:ss*:

   YYYY: Year. *MM*: Month. *DD*: Date. *hh*: Hour. *mm*: Minute. *ss*: Second.
2. Name of RDAREA subject to analysis.
3. Name of server managing the RDAREA.
4. Total number of segments in the RDAREA (number of used segments + number of unused segments).
5. Total number of unused segments in the RDAREA.
6. Segment size (number of pages per segment).
7. Page size (in bytes).
8. Name of a table or index contained in the RDAREA.
9. Authorization identifier of the table or index. For a data dictionary RDAREA, (`Data dictionary`)is displayed.
10. Percentage ratio of used segments:

    ↑ number of used segments/value of 14 ↑ × 100 (%)
11. Percentage ratio of full segments:

    ↑ number of full segments/value of 14 ↑ × 100 (%)
12. Number of used segments.
13. Number of full segments.
14. Total number of segments allocated to the table or index (number of used segments).
15. Percentage ratio of used pages:

    ↑ number of used pages/value of 19 ↑ × 100 (%)
16. Percentage ratio of full pages:

    ↑ number of full pages/value of 19 ↑ × 100 (%)
17. Number of used pages.
18. Number of full pages.
19. Total number of pages in the segments allocated to the table or index (number of used pages + number of unused pages).
20. For a table, reload-not-completed data status or being reorganized. If the table is

not in either of the following statuses, this field is blank.

`DATA_UNFINISH`:

> The table is in reload-not-completed data status, which means that table reorganization was executed, but reload processing has not been completed for some reason such as an error. To release the table from reload-not-completed data status, you must re-execute table reorganization or reload data into the table.

`ON_RORG`:

> The table is currently being reorganized.

For an index, this is the status of an unfinished index. If the index has been completed, this field is blank.

`UNFINISH_0`

> See *Figures 15-9* and *15-10*.

`UNFINISH_1`

> See *Figures 15-11* and *15-12*.

`UNFINISH_2`

> When a UAP using the plug-in index delayed batch creation function is being executed or after the UAP finished executing, a batch index creation process using the database reorganization utility is not being executed. For details about the plug-in index delayed batch creation function, see the *HiRDB Version 8 System Operation Guide*.

21. Name of the original RDAREA.
22. Generation number of the corresponding RDAREA.
23. Number of replica RDAREAs.
24. Shutdown history of the corresponding RDAREA.

    `History1`: Shutdown information immediately before the shutdown was released (if the RDAREA is currently shut down, this field displays the current shutdown information)

    `History2`: Shutdown information immediately preceding `History1`
25. Shutdown type of the corresponding RDAREA. If there is no history, this field is blank.

    `CMD`: Command shutdown by HiRDB error detection

    `FLT`: Error shutdown
26. Code indicating the cause of the shutdown of the corresponding RDAREA. If

there is no history, this field displays `0`.

`0`: Shutdown release

`10`: Input/output error

`20`: Page corruption

`40`: Open or allocation error

`70`: Rollback error

`80`: Shutdown due to an error during `DROP TABLE` or `DROP INDEX`

`90`: No-log shutdown (`pdload`, `pdrorg`, and `pdrbal` commands)

`92`: No-log shutdown (UAP)

`93`: Shutdown of LOB RDAREA

`96`: Invalid time stamp

`97`: Invalid object ID

`98`: List RDAREA error

`100`: Log application-disabled status due to execution of a utility (`pdload`, `pdrorg`, or `pdrbal`) in a mode other than the log acquisition mode

`102`: Log application-disabled status due to execution of a UAP in a mode other than the log acquisition mode

`104`: Log application-disabled status due to updating of a table for which `PARTIAL` or `NO` is specified for the BLOB column recovery restriction

`110`: Log application-disabled status due to `pdmod`

`120`: Log application-disabled status due to error shutdown

27. Shutdown time of the corresponding RDAREA. If there is no history, this field is blank.
28. Page search mode:

    `INS`: New page allocate mode

    `REU`: Free page reuse mode
29. Number of segments specified for `SEGMENT REUSE` during execution of `CREATE TABLE` or `ALTER TABLE`. If a number of segments for `SEGMENT REUSE` is not specified, this field displays `0`. If `NO` is specified for `SEGMENT REUSE` or `SEGMENT REUSE` is omitted, the field displays `-`.
30. Number of times the mode changed from *new page allocate* to *free page reuse* and then changed back to *new page allocate* because there was no reusable free space (*n*/*m*).

n: Idle count

m: Number of times mode changed from *new page allocate* to *free page reuse*

n and *m* are reset to 0 at the following times:

- When the RDAREA is closed
- When HiRDB is restarted
- When the `PURGE TABLE` statement is executed

31. Number of pages that can be released by `pdreclaim` (number of used free pages whose usage ratio is 0%).
32. Number of used pages for each 10% range. The parentheses enclose the percentage ratio to all used pages. Because the fraction part of each ratio is rounded up, the total may exceed 100. Note the following:
    - For a page that stores 201 or more duplicate keys, the percentage ratio of `PCTFREE` specified during table definition is ignored. Therefore, data is not rearranged according to the `PCTFREE` value even during reorganization.
    - For an index to which the index key value with no lock is applied, the deletion key value is treated as free space. If all used pages are deletion key values, the usage ratio is 0%.
    - A full page may not always fall in the usage ratio range of 91-100%.
    - A free space created by update processing that shortens an existing row is not treated as free space.
    - If the ratio of used pages is less than (100 - free space ratio specified with `PCTFREE`), the storage efficiency may have decreased. You should consider executing `pdrorg`. For an area storing branch rows (variable-length character string data with a length of 256 bytes or greater, data for repetition columns, and data for columns of an abstract data type), executing `pdrorg` has no effect on the storage efficiency.
33. Check pending status for referential constraint in the table information in the RDAREA. If no referential constraint has been defined for the table, this item is blank.

    `P`: Check pending status

    blank: Not check pending status
34. Check pending status for check constraint in the table information in the RDAREA. If no check constraint has been defined for the table, this item is blank.

    `P`: Check pending status

    blank: Not check pending status

Note 1

Nos. 21-23 are displayed when HiRDB Staticizer Option is used. If a generation is specified for analysis and the specified replica RDAREA generation does not exist, the utility issues a warning message, in which case no results are displayed.

Note 2

Nos. 31-32 are displayed when the `-d` option is specified.

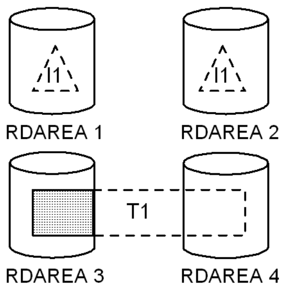
Figure 15-9: Types of unfinished index by table (1/2)

For a B-tree index

● Index unfinished status by table

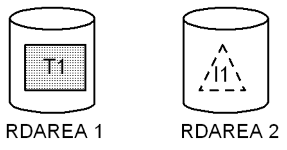
By executing batch index creation or index re-creation using `pdload` or `pdrorg`, you can change the index unfinished status by table so that the index can be used.

・Before batch creation of a partitioning key index that is subject to data loading or reorganization of a row-partitioned table by RDAREA or before batch creation of a non-partitioning key index that is partitioned within the server



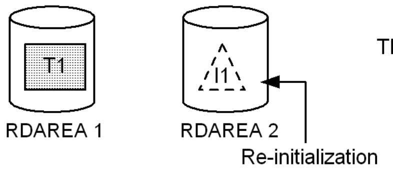
The index in RDAREA 1 corresponding to RDAREA 3 that stores the table is placed in unfinished status. The status of the index in RDAREA 2 remains unchanged.

・When `CREATE INDEX` is executed with the `EMPTY` option specified



The index in RDAREA 2 is placed in unfinished status.

・After the index storage RDAREA is re-initialized by `pdmod`

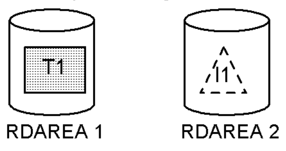


The index in RDAREA 2 is placed in unfinished status.

Figure 15-10: Types of unfinished index by table (2/2)

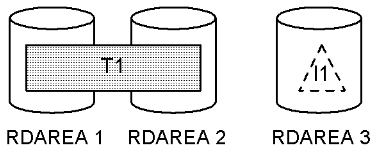
- Before batch creation of a partitioning key index or of a non-partitioning key index after data loading or reorganization by table

For a non-partitioning table or a non-partitioning key index



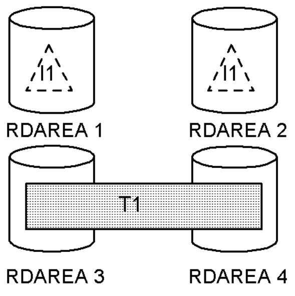
The index in RDAREA 2 is placed in unfinished status.

For a row-partitioned table or a non-partitioning key index



The index in RDAREA 3 is placed in unfinished status.

For a partitioning key index of a row-partitioned table or a non-partitioning key index for a table partitioned within the server

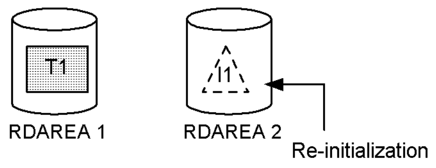


The index in RDAREAs 1 and 2 that correspond to RDAREAs 3 and 4 that store the table is placed in unfinished status.

For a plug-in index

- Re-initialized status of an index storage RDAREA placed by `pdmod`

  If an index storage RDAREA has been placed in re-initialized status by `pdmod`, you can use `pdload` or `pdrorg` to change the status of the index so that the index can be used.



The index in RDAREA 2 is placed in unfinished status.

T1 : Table

I1 : Index defined for the table

▨ or ▲ : Contains data.

⬚ or △ : Contains no data.

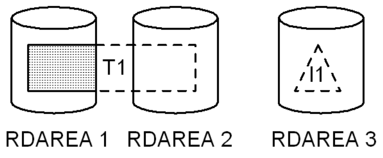
Figure 15-11: Types of unfinished index by RDAREA (1/2)

For a B-tree index

● Index unfinished status by RDAREA

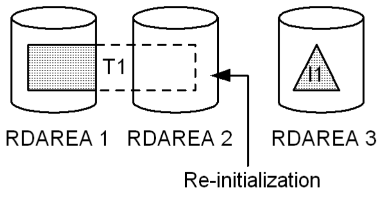
By executing batch index creation or index re-creation using `pdload` or `pdrorg`, you can change the index unfinished status by RDAREA so that the index can be used.

・Before batch creation of a non-partitioning key index that is not partitioned within the server and that is subject to data loading or reorganization of a row-partitioned table by RDAREA



The index in RDAREA 3 is placed in unfinished status.

・After partial re-initialization of a row-partitioned table for which `pdmod` was used to define the non-partitioning key index that is not partitioned within the server



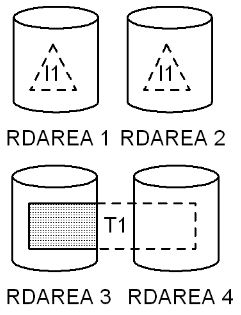
The index in RDAREA 3 is placed in unfinished status.

For a plug-in index

● Index unfinished status by table

By executing batch index creation or index re-creation using `pdload` or `pdrorg`, you can change the index unfinished status by table so that the index can be used.

・Before batch creation of a partitioning key index that is subject to data loading or reorganization of a row-partitioned table by RDAREA or before batch creation of a non-partitioning key index that is partitioned within the server

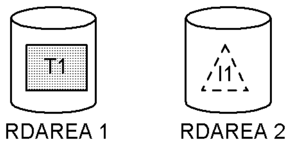


The index in RDAREA 1 corresponding to RDAREA 3 that stores the table is placed in unfinished status. The status of the index in RDAREA 2 remains unchanged.

Figure 15-12: Types of unfinished index by RDAREA (2/2)

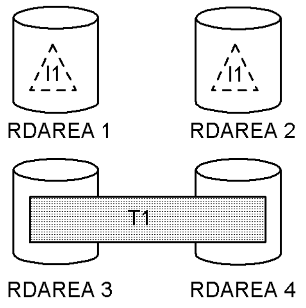
- Before batch creation of a partitioning key index or non-partitioning key index after data loading or reorganization by table

<For a non-partitioning table or non-partitioning key index>



The index in RDAREA 2 is placed in unfinished status.

<For a partitioning key index or non-partitioning key index that is partitioned within the server for a row-partitioned table>



The index in RDAREAs 1 and 2 that correspond to RDAREAs 3 and 4 containing the table is placed in unfinished status.

T1	: Table
△ I1	: Index defined for the table
▨ or ▲	: There is data.
⬚ or △	: There is no data.

(b) For LOB RDAREA

```
pddbst VV-RR(Object Option) ** RD Area Logical Analysis **  2003/04/03 12:36:43 [1]
RD Area Name   : user_rdlob_2 [2]
 Server        : bes1 [3]
 Total Segment :      11000 [4] Segment Size :          1 Pages [6]
 Unused Segment:       9178(      9178) [5]
 Page Size    :       8192 Bytes [7]
 Original RD Area Name : user_rdlob_2 [16]
 Generation Number :  0[17]  Replica RD Area Count :  0 [18]
 [19]     [20]               [21]               [22]
 History1 Hold Status :      Hold Code :     0  Hold Time :
 History2 Hold Status :      Hold Code :     0  Hold Time :
```

```
--------------------------------------------------------------------------------
 Table Name : TBL03 [8]
 Auth Id    : user1 [9]
 Status     :   [23]
          Used       Used        Sum
  Segment 100% [10]  1822 [11]  1822 [12]
  Segment Over  : N [13] Last Segment :       1822/     11000 [14]
  Lobmap  Over  : N [15]
```

Explanation

1. Date and time of completion of acquisition of the information required for condition analysis, in the format *YYYY/MM/DD hh:mm:ss*:

   YYYY: Year. *MM*: Month. *DD*: Date. *hh*: Hour. *mm*: Minute. *ss*: Second.

2. Name of RDAREA subject to analysis.
3. Name of server managing the RDAREA.
4. Total number of segments in the RDAREA (number of used segments + number of unused segments).
5. Total number of unused segments in the RDAREA. The parentheses enclose the total number of unused segments in HiRDB files that are not in the frozen update status in the corresponding RDAREA.
6. Segment size (number of pages per segment).
7. Page size (bytes).
8. Name of a table or index contained in the RDAREA.
9. Authorization identifier of the table. For a data dictionary LOB RDAREA, (`Data dictionary)` is displayed.
10. Percentage ratio of used segments:

    $\uparrow$ value of 11/value of 12 $\uparrow$ $\times$ 100 (%)

11. Number of used segments.
12. Total number of segments allocated to the table (number of used segments).
13. Whether or not the order is lost in a LOB RDAREA.

    `Y`: Order is lost.

    `N`: Order is not lost.

14. Position information indicating that the segment being used is the last segment; this is indicated in a *last position/total number of segments* format; always indicates the last segment when {13} is `Y`.

15. Whether or not the LOB management entries are all used:

    `Y`: All are used.

    `N`: There are unused entries.

16. Name of the original RDAREA.
17. Generation number of the corresponding RDAREA.
18. Number of replica RDAREAs.
19. Shutdown history of the corresponding RDAREA.

    `History1`: Shutdown information immediately before the shutdown was released (if the RDAREA is currently shut down, this field displays the current shutdown information)

    `History2`: Shutdown information immediately preceding `History1`

20. Shutdown type of the corresponding RDAREA. If there is no history, this field is blank.

    `CMD`: Command shutdown by HiRDB error detection

    `FLT`: Error shutdown

21. Code indicating the cause of the shutdown of the corresponding RDAREA. For details about the shutdown cause code, see *15.3.1(2)(a) For an RDAREA other than a LOB RDAREA*.
22. Shutdown time of the corresponding RDAREA. If there is no history, this field is blank.
23. For a table, reload-not-completed data status or being reorganized. If the table is not in either of the following statuses, this field is blank.

    `DATA_UNFINISH`:

    The table is in reload-not-completed data status, which means that table reorganization was executed, but reload processing has not been completed for some reason such as an error. To release the table from reload-not-completed data status, you must re-execute the table reorganization or reload data into the table.

    `ON_RORG`:

    The table is currently being reorganized.

    For an index, this is the status of an unfinished index. If the index has been completed, this field is blank.

    `UNFINISH_0`:

    See *Figures 15-9* and *15-10*.

`UNFINISH_1`:

See *Figures 15-11* and *15-12*.

`UNFINISH_2`:

Delayed plug-in index updating is underway. This is the status when a UAP using the plug-in index delayed batch creation function is being executed, or after the UAP finished executing, a batch index creation process using the database reorganization utility is not being executed. For details about the plug-in index delayed batch creation function, see the *HiRDB Version 8 System Operation Guide*.

Note

Nos. 16-18 are displayed when HiRDB Staticizer Option is used. If a generation is specified for analysis and the specified replica RDAREA generation does not exist, the utility issues a warning message, in which case no results are displayed.

(3) Analyzing the analysis results

The results of condition analysis by RDAREA (logical analysis) can be analyzed as shown in Table 15-1.

Table 15-1: Analyzing the analysis results by RDAREA (logical analysis)

Status in the RDAREA	Description
Percentage of total unused pages in the table is less than the percentage of free pages per segment that was specified in the `CREATE TABLE` definition SQL statement.	Table data may be in one of the following statuses: • There are only a few unused pages because data was added repeatedly. • Used free pages are scattered because data addition, deletion, and updating occurred repeatedly. If necessary, you can correct the table arrangement by expanding the RDAREA that stores the table and then reorganizing the table.[1, 2]
The total number of used index storage pages displayed in the analysis results is greater than the number of index storage pages obtained from the number of table storage rows.	The index may be in the following status: • Because many rows in a specific range were deleted, a large amount of the index was also deleted, resulting in many used free pages. In this case, you can correct the index arrangement by reorganizing the index. You can adjust the number of used pages to a reasonable value by releasing free index pages. For details about determining the number of index storage pages, see the *HiRDB Version 8 Installation and Design Guide*.

Status in the RDAREA			Description
Table with clustering	Used free pages	Many	Percentage of free pages per segment specified in `CREATE TABLE` is too large. If necessary, reorganize the table because used free pages have been created due to data addition, deletion, and updating.[1]
		Few	If you are adding data, the table should be reorganized.[1]
Table without clustering	Used free pages	Many	Percentage of free pages per segment specified in `CREATE TABLE` is too large. If necessary, reorganize the table because used free pages have been created due to data addition, deletion, and updating.[1]
		Few	If there are only a few unused segments in the RDAREA, expand the RDAREA as required.[2]
End-of-segment is Y			`LOB` column search performance may become poor. Reorganize the LOB RDAREA.[1]
End-of-Lobmap is Y			`LOB` column search performance may become poor. Reorganize the `LOB` column structure base table.[1]

Note

If an RDAREA storing the LOB attribute of a plug-in-provided abstract data type or an RDAREA storing a plug-in index is analyzed, the percentage of used segments in the RDAREA depends on the method used by the plug-in to allocate segments. For example, if a plug-in completes allocation of segments during index definition, most of the segments may become used segments immediately upon definition (before data is stored).

[1] For details about table and index reorganization, see Chapter *8. Database Reorganization Utility (pdrorg)*. For details about releasing free index pages, see Chapter *11. Free Page Release Utility (pdreclaim)*.

[2] For details about RDAREA expansion, see Chapter *7. Database Structure Modification Utility (pdmod)*.

15.3.2 Condition analysis by RDAREA (physical analysis)

The utility analyzes the storage condition of all segments and all pages in an RDAREA without taking into account tables or indexes.

(1) Purpose

The utility displays the storage condition of all segments and all pages in an RDAREA. From the displayed analysis results, you can determine the RDAREA's utilization efficiency and capacity status.

(2) Analysis results

The following shows the results of condition analysis by RDAREA (physical analysis):

- In text format (`-a` option is omitted)

```
 pddbst VV-RR(Object Option) ** RD Area Physical Analysis ** 2002/03/29 16:40:10 [1]
  Max Sum Segment :        150 [2]
  Max Sum Page    :        750 [3]
  RD Area Count   :          1/         1 [4]

--------------------------------------------------------------------------------
 RD Area Name   : USER_RDAREA_1 [5]
  Server        : BES01 [6]
  Original RD Area Name : USER_RDAREA_1 [30]
  Generation Number :  0[31] Replica RD Area Count :  0[32]
    [33]                [34]                [35]            [36]
  History1 Hold Status :      Hold Code :     0  Hold Time :
  History2 Hold Status :      Hold Code :     0  Hold Time :
  Unused Segment:         98 [7]
  Segment Size  :          5 Pages[8] Page Size    :       4096 Bytes[9]
          Used(Full)       Used(      Full)        Sum
           [10] [11]        [12]       [13]       [14]
  Segment  35%(  0%)          52(        0)        150
           [15] [16]        [17]       [18]       [19]
  Page     35%(  0%)         256(        0)        750

  Auto Extend Use    : USE[20]   Auto Extend Segment :     4 [21]
  Auto Extend Status : NOSUP[22] Error Code          :     0 [23]
   HiRDB File Name : /tmp/pddir/ios_area01/u_rd20_1 [24]
    File Size      :        5 segments[25] Extent Count :   1/ 24 [26]
   HiRDB File Name : /tmp/pddir/ios_area01/u_rd20_2
    File Size      :        6 segments     Extent Count :   1/ 24
   HiRDB File Name : /tmp/pddir/ios_area01/u_rd20_3
    File Size      :       18 segments     Extent Count :   1/ 24
```

```
RD Area Name   : USER_RDLOB_1
 Server        : BES01
 Original RD Area Name : USER_RDLOB_1
 Generation Number :  0   Replica RD Area Count :  0
 History1 Hold Status :      Hold Code :     0  Hold Time :
 History2 Hold Status :      Hold Code :     0  Hold Time :
 Unused Segment:        770(       770)
 Segment Size  :          1 Pages   Page Size    :       8192 Bytes
        Used       Used        Sum
 Segment  93%      10230      11000
              [27]                         [28]
 Segment Over  : N  Last Segment :      10230/     11000
              [29]
 Lobmap  Over  : N
 Freeze Specified : Y [37]

 Auto Extend Use    : USE     Auto Extend Segment :    10
 Auto Extend Status : NOSUP   Error Code          :     0
  HiRDB File Name : /tmp/pddir/ios_area01/u101
   File Size      :       4000 segments   Extent Count :   1/ 24
   Segment        :       4000/      4000
   Freeze Status  : U[38] ****/**/**    **:**:** [39]
  HiRDB File Name : /tmp/pddir/ios_area01/u201
   File Size      :       5000 segments   Extent Count :   1/ 24
   Segment        :       5000/      5000
   Freeze Status  : F    2002/03/29    16:38:19
  HiRDB File Name : /tmp/pddir/ios_area01/u301
   File Size      :       2000 segments   Extent Count :   1/ 24
   Segment        :       1230/      2000
   Freeze Status  : U    ****/**/**    **:**:**
```

Explanation

RDAREA information is displayed in ascending order of the RDAREA names. HiRDB file information is displayed in the order of the physical file IDs.

1. Date and time of completion of acquisition of the information required for this condition analysis, in the format *YYYY/MM/DD hh:mm:ss*:

   YYYY: Year. *MM*: Month. *DD*: Date. *hh*: Hour. *mm*: Minute. *ss*: Second.

2. Maximum total number of segments in the analyzed RDAREAs.
3. Maximum total number of pages in the analyzed RDAREAs.
4. Number of specified RDAREAs and number of RDAREAs analyzed successfully.
5. Name of an analyzed RDAREA.
6. Name of the server managing the RDAREA.
7. Total number of unused segments in the RDAREA. The parentheses enclose the total number of unused segments in the HiRDB files that are not in the frozen update status in the corresponding RDAREA.

8. Segment size (number of pages per segment).
9. Page size (in bytes).
10. Percentage ratio of used segments:

    ↑ number of used segments/value of 14 ↑ × 100 (%)
11. Percentage ratio of full segments:

    ↑ number of full segments/value of 14 ↑ × 100 (%)
12. Number of used segments.
13. Number of full segments.
14. Total number of segments in the RDAREA (number of used segments + number of unused segments).
15. Percentage ratio of used pages:

    ↑ number of used pages/value of 19 ↑ × 100 (%)
16. Percentage ratio of full pages:

    ↑ number of full pages/value of 19 ↑ × 100 (%)
17. Number of used pages.
18. Number of full pages.
19. Total number of pages in the segments in the RDAREA (number of used pages + number of unused pages).
20. Utilization status of the automatic extension facility:

    `USE`: Automatic extension facility is used.

    `NOUSE`: Automatic extension facility is not used.

    The HiRDB file indicated as the last `HiRDB File Name` is subject to automatic extension.
21. Number of extension segments.
22. Suppression status of the automatic extension facility:

    `SUP`: Suppressed.

    `NOSUP`: Not suppressed
23. Error code in the event of an automatic extension error.

    `-1556`:

    Take one of the following actions:

- Use the `pdfbkup`, `pdfmkfs`, and `pdfrstr` commands to integrate the extents of the HiRDB file system area that contains the corresponding HiRDB file system.
- Use `pdmod` to expand the RDAREA.

Other:

See the list of HiRDB file system error codes in the manual *HiRDB Version 8 Messages*.

24. HiRDB file name in the corresponding RDAREA.
25. Total number of segments for the HiRDB file.
26. Number of HiRDB file segments stored/maximum number of segments.
27. Whether or not the order is lost in a LOB RDAREA:

    `Y`: Order is lost.

    `N`: Order is not lost.
28. Position information indicating that the segment being used is the last segment; this is indicated in a *last position/total number of segments* format; always indicates the last segment when {27} is `Y`.
29. Whether or not the LOB management entries are all used:

    `Y`: All are used.

    `N`: There are unused entries.
30. Name of the original RDAREA.
31. Generation number of the corresponding RDAREA.
32. Number of replica RDAREAs.
33. Shutdown history of the corresponding RDAREA.

    `History1`: Shutdown information immediately before the shutdown was released (if the RDAREA is currently shut down, this field displays the current shutdown information)

    `History2`: Shutdown information immediately preceding `History1`
34. Shutdown type of the corresponding RDAREA. If there is no history, this field is blank.

    `CMD`: Command shutdown by HiRDB error detection

    `FLT`: Error shutdown
35. Code indicating the cause of the shutdown of the corresponding RDAREA. For details about the shutdown cause code, see *15.3.1(2)(a) For an RDAREA other*

than a LOB RDAREA.

36. Shutdown time of the corresponding RDAREA. If there is no history, this field is blank.

37. Whether or not the RDAREA is in frozen update status:

    `Y`: Placed in frozen update status by the `pddbfrz` command.

    `N`: Not placed in frozen update status by the `pddbfrz` command.

38. Whether or not the HiRDB files constituting the RDAREA are in frozen update status:

    `U`:

    > Not in frozen update status. During backup, such HiRDB files need to be backed up.

    `F`:

    > In the frozen update status. If a backup was made subsequent to the displayed date and time, there is no need to back up these HiRDB files again.

39. Date and time the frozen update status was changed.

    When `****/**/** **:**:**` is displayed:

    > This information is displayed if the RDAREA has never been placed in frozen update status since it was created or re-initialized.

    Other that the above:

    > The displayed information indicates the date and time the RDAREA was placed in frozen update status or released from frozen update status.

Note 1

Nos. 10-11 and 15-19 are not displayed for a LOB RDAREA.

Note 2

Nos. 27-29 and 37 are displayed for a LOB RDAREA.

Note 3

Nos. 30-32 are displayed when HiRDB Staticizer Option is used. If a generation is specified for analysis and the specified replica RDAREA generation does not exist, the utility issues a warning message, in which case no results are displayed.

- In DAT format (when the `-a` option is specified)

  Table 15-2 describes the results of condition analysis by RDAREA (physical analysis).

  Rules for output in DAT format are as follows:

- The items described in Table 15-2 are separated by the comma (`,`).
- Information about one RDAREA is displayed per line. If a single RDAREA consists of multiple HiRDB files, the information for the RDAREA consists of as many lines as there are component HiRDB files.
- RDAREA information is displayed in ascending order of the RDAREA names. HiRDB file information is displayed in the order of the physical file IDs.

Table 15-2: Results of condition analysis by RDAREA (physical analysis) in DAT format

Title bar[1]	**Description**	**Output format**	**Max. length (bytes)**[2]
`VERSION`	HiRDB version	Character string (*vv*-*rr*)	5
`INF_GET_TIME`	Date (year, month, date) and time that information acquisition was completed	Character string (*yyyy*/*mm*/*dd* *hh*:*mm*:*ss*)	19
`RDAREA_NAME`	Name of the RDAREA	Character string	30
`SERVER`	Name of the server containing the RDAREA	Character string	8
`MAX_SUM_SEGMENT`	Maximum total number of segments in the RDAREA	Numeric value	10
`MAX_SUM_PAGE`	Maximum total number of pages in the RDAREA	Numeric value	10
`SPECIFIED_RDAREA_COUNT`	Number of RDAREAs to be analyzed	Numeric value	10
`ANALYZED_RDAREA_COUNT`	Number of RDAREAs analyzed normally	Numeric value	10
`SEGMENT_SIZE`	Number of pages per segment	Numeric value	10
`PAGE_SIZE`	Page size	Numeric value	10
`HOLD_STATUS(HISTORY1)`[5]	RDAREA shutdown type (type of the last shutdown that was released; or if RDAREA is currently shut down, the current shutdown type): `CMD`: Command shutdown by HiRDB error detection `FLT`: Error shutdown	Character string	3

Title bar[1]	Description	Output format	Max. length (bytes)[2]
`HOLD_CODE(HISTORY1)`	RDAREA shutdown cause code (cause code of the last shutdown that was released, or if RDAREA is currently shut down, the current shutdown cause code) (For details about the shutdown cause code, see the description for the text format.)	Numeric value	5
`HOLD_TIME(HISTORY1)`[5]	RDAREA shutdown time (time of the last shutdown that was released; or if RDAREA is currently shut down, the current shutdown time)	Character string (*yyyy*/*mm*/*dd* *hh*:*mm*:*ss*)	19
`HOLD_STATUS(HISTORY2)`[5]	RDAREA shutdown type (shutdown type immediately preceding History1): `CMD`: Command shutdown `FLT`: Error shutdown	Character string	3
`HOLD_CODE(HISTORY2)`	RDAREA shutdown cause code (shutdown cause code immediately preceding History1) (For details about the shutdown cause code, see the description for the text format.)	Numeric value	5
`HOLD_TIME(HISTORY2)`[5]	RDAREA shutdown time (shutdown time immediately preceding History1)	Character string (*yyyy*/*mm*/*dd* *hh*:*mm*:*ss*)	19
`FREEZE_SPECIFIED`[3]	Whether or not frozen update was specified for the RDAREA: `Y`: Frozen update was specified with the `pddbfrz` command `N`: Frozen update was not specified with the `pddbfrz` command	Character string	1
`UNUSED_SEGMENT`	Total number of unallocated segments in the RDAREA	Numeric value	10
`UNUSED_SEGMENT_NOT_FREEZE`[3]	Total number of unallocated segments in HiRDB files that are not in frozen update status in the RDAREA	Numeric value	10
`USED_SEGMENT`	Number of used segments	Numeric value	10

Title bar[1]	Description	Output format	Max. length (bytes)[2]
`USED_SEGMENT_RATIO`	Percentage of used segments ↑(number of used segments/total number of segments in RDAREA) × 100↑	Numeric value	3
`FULL_SEGMENT`	Number of full segments	Numeric value	10
`FULL_SEGMENT_RATIO`	Percentage of full segments ↑(number of full segments/total number or segments in RDAREA) × 100↑	Numeric value	3
`SUM_SEGMENT`	Total number of segments in the RDAREA	Numeric value	10
`SEGMENT_OVER`[3]	Whether or not all segments are in use or have been used: `Y`: Used `N`: Not used	Character string	1
`LAST_SEGMENT_IN_USE`[3]	Last segment in use (if end-of-segment is `Y`, the last segment)	Numeric value	10
`TOTAL_SEGMENT_ NUMBER`[3]	Total number of segments	Numeric value	10
`LOBMAP_OVER`[3]	Whether or not all LOB management entries have been used: `Y`: Used `N`: Not used	Character string	1
`USED_LOB_ MANAGEMENT_ENTRY_NUMBER`	Information used by the system	Numeric value	10
`TOTAL_LOB_ MANAGEMENT_ENTRY_NUMBER`	Information used by the system	Numeric value	10
`USED_PAGE`	Number of used pages	Numeric value	10
`USED_PAGE_RATIO`	Percentage of used pages ↑(number of used pages/total number of pages in all segments in RDAREA) × 100↑	Numeric value	3
`FULL_PAGE`	Number of full pages	Numeric value	10

Title bar[1]	Description	Output format	Max. length (bytes)[2]
`FULL_PAGE_RATIO`	Percentage of full pages ↑(number of full pages/total number of pages in all segments in RDAREA) $\times$ 100↑	Numeric value	3
`SUM_PAGE`	Total number of pages in all segments in the RDAREA	Numeric value	10
`AUTO_EXTEND_USE`	Usage status of automatic extension facility: `USE`: Used `NOUSE`: Not used The last HiRDB file displayed in the HiRDB file information uses the automatic extension facility.	Character string	5
`AUTO_EXTEND_SEGMENT`	Number of extended segments	Numeric value	5
`AUTO_EXTEND_STATUS`	Automatic extension facility suppression status: `SUP`: Suppressed `NOSUP`: Not suppressed	Character string	5
`ERROR_CODE`	Error code for an automatic extension error	Numeric value	5
`HiRDB_FILE_NAME`	HiRDB file name in the RDAREA (absolute path name)	Character string	167
`FILE_SIZE`	HiRDB file size (total number of segments)	Numeric value	10
`EXTENT_COUNT`	HiRDB file partition storage count	Numeric value	3
`MAX_EXTENT_COUNT`	Maximum HiRDB file partition storage count value	Numeric value	3
`USED_SEGMENT_IN_FILE`[3]	Number of used segments in HiRDB file	Numeric value	10
`TOTAL_SEGMENT_IN_FILE`[3]	Total number of segments in HiRDB file	Numeric value	10

Title bar[1]	Description	Output format	Max. length (bytes)[2]
`FREEZE_STATUS`[3]	Frozen update status of HiRDB file: `U`: Not in frozen update status (when a backup is made, this HiRDB file must be backed up) `F`: In frozen update status (if the HiRDB file was backed up after the displayed date and time, there is no need to back it up again)	Character string	1
`FREEZE_TIME`[3]	Frozen update status change date and time: `****/**/** **:**:**`: The file has not been placed in frozen update status since the RDAREA was created or reinitialized Other: Date and time that frozen update status was set or released	Character string (*yyyy*/*mm*/*dd* *hh*:*mm*:*ss*)	19
`ORIGINAL_RDAREA_NAME`[4]	Original RDAREA name	Character string	30
`GENERATION_NUMBER`[4]	Generation number of the RDAREA	Numeric value	2
`REPLICA_RDAREA_COUNT`[4]	Number of replica RDAREAs	Numeric value	2

[1] The title bar is displayed when the `-h` option is specified.

[2] When the output format is character string, this information is enclosed in double quotation marks (`""`), in which case the maximum length does not include the double quotation marks. When the output format is not character string, the information is not enclosed in double quotation marks.

[3] This information is displayed only for a LOB RDAREA. For any other type of RDAREA, no data is output (example: `...,,...`).

[4] This information is displayed only when HiRDB Staticizer Option is used. When HiRDB Staticizer Option is not used, no data is displayed (example: `...,,...`).

[5] If the RDAREA has never been shut down, no data is output (example: `...,,...`).

(3) Analyzing the analysis results

The results of condition analysis by RDAREA (physical analysis) can be analyzed as shown in Table 15-3.

Table 15-3: Analyzing the analysis results by RDAREA (physical analysis)

Status in the RDAREA					Description
Entire RDAREA	Unused segments	Many			There is sufficient capacity.
		Few	Used free pages	Many	The RDAREA is not being used efficiently. Execute condition analysis by RDAREA (logical analysis), check for a table or index that has many used segments and unused pages; if necessary, reorganize such a table or index or release free pages.[1]
				Few	There is not enough capacity. Extend the RDAREA.[2]

[1] For details about table and index reorganization, see Chapter *8. Database Reorganization Utility (pdrorg)*. For details about releasing free table and index pages, see Chapter *11. Free Page Release Utility (pdreclaim)*.

[2] For details about RDAREA expansion, see Chapter *7. Database Structure Modification Utility (pdmod)*.

A user LOB RDAREA can be analyzed as follows:

- Information about segments in the RDAREA

Percentage of used segments	End-of-segment	Description
Large (there are only a few unused segments)	Y	There is not enough capacity and database access performance has been affected adversely. Expand the user LOB RDAREA and reorganize the table or index.[1, 2]
	N	There is not enough capacity. The user LOB RDAREA must be extended.[1]
Small (there are many unused segments)	Y	Database access performance has been affected adversely. Reorganize the table or index.[2]
	N	No action is necessary.

Note

If an RDAREA storing the LOB attribute of a plug-in-provided abstract data type or an RDAREA storing a plug-in index is analyzed, the percentage of used segments in the RDAREA depends on the method used by the plug-in to allocate segments. For example, if a plug-in completes allocation of segments during index definition, most of the segments may become used segments immediately upon definition (before data is stored).

1 For details about RDAREA expansion, see Chapter *7. Database Structure Modification Utility (pdmod)*.

2 For details about table and index reorganization, see Chapter *8. Database Reorganization Utility (pdrorg)*.

- End-of-Lobmap

  If this is Y, LOB column search performance may become poor; you should reorganize the LOB column structure base table.

15.3.3 Condition analysis by table or index

The utility analyzes the segment status of a table or index storage page and the status of all storage pages by RDAREA.

(1) Purpose

You can determine the degree of disorganization of tables or indexes. For a table or index that has been partitioned and stored in multiple RDAREAs, you can determine whether or not partitions are distributed uniformly among the RDAREAs.

(2) Analysis results

The following shows the results of condition analysis by table or index:

(a) Condition analysis by table

- When no LOB RDAREA is included

```
 pddbst VV-RR(Object Option) ** Table Analysis **            2003/03/31 18:53:24 [1]
 Table Name     : TBL01 [2]
 Auth Id        : user1 [3]
 Hash           :   [4]
  Total Segment :        104 [5] Max Sum Segment :        104 [7]
  Total Page    :        520 [6] Max Sum Page    :        520 [8]
  Total Row     :                   500 [9]
  RD Area Count :          1/         1 [10]


--------------------------------------------------------------------------------
 RD Area Name   : user_rdarea_1 [11]
  Server        : bes1 [12]
  Reference Pending Status : P [41]
  Check     Pending Status : P [42]
  Original RD Area Name : user_rdarea_1 [28]
  Generation Number :  0[29] Replica RD Area Count :  0[30]
  Status        :   [38]
  [31]     [32]               [33]               [34]
  History1 Hold Status :      Hold Code :     0  Hold Time :
  History2 Hold Status :      Hold Code :     0  Hold Time :
  Job Name      :   [25]  Line Count :   [26]         Index Method :   [27]
```

```
Unused Segment:         24 [13]
Search Mode : INS [35] Segment Reuse :          - segments [36]
Reuse Search Failure :          0/         0 [37]
        Used(Full)       Used(      Full)        Sum
        [14]  [15]        [16]       [17]       [18]
Segment 100%(  0%)         104(         0)        104
        [19]  [20]         [21]       [22]       [23]
Page    100%(  0%)         518(         0)        520
Row Count     :                     500 [24]

Collect Prearranged Page :          55 [39]
Used Page Ratio       Page(Ratio) [40]
           0% :         57(  11%)
       1- 10% :         19(   4%)
      11- 20% :         28(   6%)
      21- 30% :         70(  14%)
      31- 40% :         95(  19%)
      41- 50% :         71(  14%)

      51- 60% :         69(  14%)
      61- 70% :         58(  12%)
      71- 80% :         37(   8%)
      81- 90% :         14(   3%)
      91-100% :          2(   1%)
Total                  520
```

Explanation

1. Date and time of completion of acquisition of the information required for this condition analysis, in the format *YYYY/MM/DD hh:mm:ss*:

   YYYY: Year. *MM*: Month. *DD*: Date. *hh*: Hour. *mm*: Minute. *ss*: Second.

2. Name of the table subject to analysis.
3. Authorization identifier of the table subject to analysis.
4. Name of the hash function used if the table is partitioned with flexible hash or FIX hash. `Hash` is displayed for flexible hash partitioning; `Fix Hash` is displayed for FIX hash partitioning. This field is blank if the table uses neither flexible nor FIX hash partitioning.
5. Total number of segments allocated to the table (number of used segments + number of unused segments).
6. Total number of pages in the segments allocated to the table (used segments + unused segments).
7. Maximum total number of segments allocated to the table per RDAREA.
8. Maximum total number of pages in the segments allocated to the table per RDAREA.
9. Total number of rows stored in the table (total number of rows stored in each

RDAREA); displayed only when the -s option is specified.

10. Number of RDAREAs analyzed successfully and number of RDAREAs containing the table storage pages.
11. Name of RDAREA subject to analysis.
12. Indicates the name of the server managing the RDAREA.
13. Total number of unused segments in the RDAREA. The parentheses enclose the total number of unused segments in HiRDB files that are not in the frozen update status in the corresponding RDAREA.
14. Percentage ratio of used segments:

    ↑ number of used segments/value of 18 ↑ × 100 (%)
15. Percentage ratio of full segments:

    ↑ number of full segments/value of 18 ↑ × 100 (%)
16. Number of used segments.
17. Number of full segments.
18. Total number of segments allocated to the corresponding table in the RDAREA (number of used segments).
19. Percentage ratio of used pages:

    ↑ number of used pages/value of 23 ↑ × 100 (%)
20. Percentage ratio of full pages:

    ↑ number of full pages/value of 23 ↑ × 100 (%)
21. Number of used pages.
22. Number of full pages.
23. Total number of pages in the segments allocated to the corresponding table in the RDAREA (number of used pages + number of unused pages).
24. Number of rows stored in the RDAREA; displayed only when the `-s` option is specified. Because `NOWAIT` retrieval is performed to count the number of stored rows, errors may occur.
25. Job name specified during data loading, reorganization, or reloading with the synchronization point specification. Parentheses enclose the type of utility that uses this job:

    `L`: `pdload`

    `R`: `pdrorg`

blank: Not set

26. Number of rows stored in the database during data loading, reorganization, or reloading with the synchronization point specification.
27. Index creation method used during data loading, reorganization, or reloading with the synchronization point specification (value of the `-i` option).
28. Name of the original RDAREA.
29. Generation number of the corresponding RDAREA.
30. Number of replica RDAREAs.
31. Shutdown history of the corresponding RDAREA.

    `History1`: Shutdown information immediately before the shutdown was released (if the RDAREA is currently shut down, this field displays the current shutdown information)

    `History2`: Shutdown information immediately preceding `History1`
32. Shutdown type of the corresponding RDAREA. If there is no history, this field is blank.

    `CMD`: Command shutdown by HiRDB error detection

    `FLT`: Error shutdown
33. Code indicating the cause of the shutdown of the corresponding RDAREA. For details about the shutdown cause code, see *15.3.1(2)(a) For an RDAREA other than a LOB RDAREA*.
34. Shutdown time of the corresponding RDAREA. If there is no history, this field is blank.
35. Page search mode:

    `INS`: New page allocate mode

    `REU`: Free page reuse mode
36. Number of segments specified for `SEGMENT REUSE` during execution of `CREATE TABLE` or `ALTER TABLE`. If a number of segments for `SEGMENT REUSE` is not specified, this field displays `0`. If `NO` is specified for `SEGMENT REUSE` or `SEGMENT REUSE` is omitted, the field displays `-`.
37. Number of times the mode changed from *new page allocate* to *free page reuse* and then changed back to *new page allocate* because there was no reusable free space (*n*/*m*).

    n: Idle count

    m: Number of times mode changed from *new page allocate* to *free page reuse* *n* and *m* are reset to 0 at the following times:

- When the RDAREA is closed
- When HiRDB is restarted
- When the `PURGE TABLE` statement is executed

38. Whether the table is in reload-not-completed data status or is being reorganized. If the table is not in either of the following statuses, this field is blank.

    `DATA_UNFINISH`:

    The table is in reload-not-completed data status, which means that table reorganization was executed, but reload processing has not been completed for some reason such as an error. To release the table from reload-not-completed data status, you must re-execute the table reorganization or reload data into the table.

    `ON_RORG`:

    The table is currently being reorganized.

39. Number of pages that can be released by `pdreclaim` (number of used free pages whose usage ratio is 0%).
40. Number of used pages for each 10% range. The parentheses enclose the percentage ratio to all used pages. Because the fraction part for each ratio is rounded up, the total may exceed 100. Note the following:
    - A full page may not always fall in the usage ratio range of 91-100%.
    - A free space created by update processing that shortens an existing row is not treated as free space.
    - If the ratio of used pages is less than (100 - free space ratio specified with `PCTFREE`), the storage efficiency may have decreased. You should consider executing `pdrorg`. For an area storing branch rows (variable-length character string data with a length of 256 bytes or greater, data for repetition columns, and data for columns of abstract data type), executing `pdrorg` has no effect on the storage efficiency.
41. Check pending status for referential constraint in the table information in the RDAREA. If no referential constraint has been defined for the table, this item is blank.

    `P`: Check pending status

    blank: Not check pending status
42. Check pending status for check constraint in the table information in the RDAREA. If no check constraint has been defined for the table, this item is blank.

`P`: Check pending status

blank: Not check pending status

Note 1

Nos. 28-30 are displayed when HiRDB Staticizer Option is used. If a generation is specified for analysis and the specified replica RDAREA generation does not exist, the utility issues a warning message, in which case no results are displayed.

Note 2

Nos. 39-40 are displayed when the `-d` option is specified.

- When a LOB RDAREA is included

```
 pddbst VV-RR(Object Option) ** Table Analysis **           2003/04/03 12:37:22 [1]
 Table Name      : TBL03 [2]
 Auth Id         : user1 [3]
 Hash            :   [4]
  Total Segment :       1826 [5] Max Sum Segment :       1822 [7]
  Total Page    :       1842 [6] Max Sum Page    :       1822 [8]
  Total Row     :                   500 [9]
  RD Area Count :          2/         2 [10]

--------------------------------------------------------------------------------
 RD Area Name    : user_rdarea_3 [11]
  Server         : bes1 [12]
  Status         :   [42]
  Original RD Area Name : user_rdarea_3 [32]
  Generation Number :  0[33] Replica RD Area Count :  0[34]
  [35]     [36]              [37]               [38]
  History1 Hold Status :      Hold Code :     0  Hold Time :
  History2 Hold Status :      Hold Code :     0  Hold Time :
  Job Name       :   [29]  Line Count :   [30]         Index Method :   [31]

  Unused Segment:        145 [13]
  Search Mode : INS [39] Segment Reuse :          - segments [40]
  Reuse Search Failure :         0/         0 [41]
          Used(Full)       Used(      Full)        Sum
          [14]  [15]        [16]       [17]        [18]
  Segment 100%(  0%)          4(         0)          4
          [19]  [20]        [21]       [22]        [23]
  Page     85%(  0%)         17(         0)         20
  Row Count     :                    500 [24]


 RD Area Name    : user_rdlob_2
  Server         : bes1
  Status         :
  Column Name    : C2 [25]
  Original RD Area Name : user_rdlob_2
  Generation Number :  0     Replica RD Area Count :  0
  History1 Hold Status :      Hold Code :     0  Hold Time :
  History2 Hold Status :      Hold Code :     0  Hold Time :
```

```
Unused Segment:       9178(      9178)
        Used       Used        Sum
Segment 100%       1822       1822
Segment Over   : N  [26] Last Segment :       1822/     11000 [27]
Lobmap  Over   : N [28]
```

Explanation

1. Date and time of completion of acquisition of the information required for this condition analysis, in the format *YYYY*/*MM*/*DD* *hh*:*mm*:*ss*:

   YYYY: Year. *MM*: Month. *DD*: Date. *hh*: Hour. *mm*: Minute. *ss*: Second.

2. Name of the table subject to analysis.
3. Authorization identifier of the table subject to analysis.
4. Name of the hash function used if the table is partitioned with flexible hash or FIX hash. `Hash` is displayed for flexible hash partitioning; `Fix Hash` is displayed for FIX hash partitioning. This field is blank if the table uses neither flexible nor FIX hash partitioning.
5. Total number of segments allocated to the table (number of used segments + number of unused segments).
6. Total number of pages in the segments allocated to the table (used segments + unused segments).
7. Maximum total number of segments allocated to the table per RDAREA.
8. Maximum total number of pages in the segments allocated to the table per RDAREA.
9. Total number of rows stored in the table (total number of rows stored in each RDAREA). This information is displayed only when the `-s` option is specified.
10. Number of RDAREAs analyzed successfully and number of RDAREAs containing the table storage pages.
11. Name of RDAREA subject to analysis. If it is a user LOB RDAREA, this field displays the name of the user RDAREA for the LOB column structure base table, followed by the name of the user LOB RDAREA for the LOB data.
12. Indicates the name of the server managing the RDAREA.
13. Total number of unused segments in the RDAREA. The parentheses enclose the total number of unused segments in HiRDB files that are not in frozen update status in the corresponding RDAREA.
14. Percentage ratio of used segments:

    ↑ number of used segments/value of 18 ↑ $\times$ 100 (%)

15. Percentage ratio of full segments:

    ↑ number of full segments/value of 18 ↑ × 100 (%)
16. Number of used segments.
17. Number of full segments.
18. Total number of segments allocated to the corresponding table in the RDAREA (number of used segments).
19. Percentage ratio of used pages:

    ↑ number of used pages/value of 23 ↑ × 100 (%)
20. Percentage ratio of full pages:

    ↑ number of full pages/value of 23 ↑ × 100 (%)
21. Number of used pages.
22. Number of full pages.
23. Total number of pages in the segments allocated to the corresponding table in the RDAREA (number of used pages + number of unused pages).
24. Number of rows stored in the RDAREA. This information is displayed only when the `-s` option is specified. There may be an error in the storage rows count because a `NOWAIT` search is conducted.
25. Name of LOB column.
26. Whether or not order has been lost in the LOB RDAREA:

    `Y`: Order has been lost.

    `N`: Order has not been lost.
27. Position information indicating that the segment being used is the last segment. This information is displayed in the format *last-position/ total-number-of-segments*. When No. 26 is `Y`, this field always indicates the last segment.
28. Whether or not the LOB management entries are all used:

    `Y`: All are used.

    `N`: There are unused entries.
29. Job name specified during data loading, reorganization, or reloading with the synchronization point specification. Parentheses enclose the type of utility that uses this job:

    `L`: `pdload`

R: `pdrorg`

blank: Not set

30. Number of rows stored in the database during data loading, reorganization, or reloading with the synchronization point specification.
31. Index creation method used during data loading, reorganization, or reloading with the synchronization point specification (value of the `-i` option).
32. Name of the original RDAREA.
33. Generation number of the corresponding RDAREA.
34. Number of replica RDAREAs.
35. Shutdown history of the corresponding RDAREA.

    `History1`: Shutdown information immediately before the shutdown was released (if the RDAREA is currently shut down, this field displays the current shutdown information)

    `History2`: Shutdown information immediately preceding `History1`
36. Shutdown type of the corresponding RDAREA. If there is no history, this field is blank.

    `CMD`: Command shutdown by HiRDB error detection

    `FLT`: Error shutdown
37. Code indicating the cause of the shutdown of the corresponding RDAREA. For details about the shutdown cause code, see *15.3.1(2)(a) For an RDAREA other than a LOB RDAREA*.
38. Shutdown time of the corresponding RDAREA. If there is no history, this field is blank.
39. Page search mode:

    `INS`: New page allocate mode

    `REU`: Free page reuse mode
40. Number of segments specified for `SEGMENT REUSE` during execution of `CREATE TABLE` or `ALTER TABLE`. If a number of segments for `SEGMENT REUSE` is not specified, this field displays `0`. If `NO` is specified for `SEGMENT REUSE` or `SEGMENT REUSE` is omitted, the field displays `-`.
41. Number of times the mode changed from *new page allocate* to *free page reuse* and then changed back to *new page allocate* because there was no reusable free space (*n*/*m*).

    n: Idle count

m: Number of times mode changed from *new page allocate* to *free page reuse*

n and *m* are reset to 0 at the following times:

- When the RDAREA is closed
- When HiRDB is restarted
- When the `PURGE TABLE` statement is executed

42. Whether the table is in reload-not-completed data status or is being reorganized. If the table is not in either of the following statuses, this field is blank.

    `DATA_UNFINISH`:

    The table is in reload-not-completed data status, which means that table reorganization was executed, but reload processing has not been completed for some reason such as an error. To release the table from reload-not-completed data status, you must re-execute table reorganization or reload data into the table.

    `ON_RORG`:

    The table is currently being reorganized.

Note

Nos. 32-34 are displayed when HiRDB Staticizer Option is used. If a generation is specified for analysis and there is no specified generation of replica RDAREA area, the utility issues a warning message, in which case no results are displayed.

(b) Condition analysis by index

■ B-tree index

```
pddbst VV-RR(Object Option) ** Index Analysis **            2003/03/31 18:56:00 [1]
Index Name      : IDX_TBL01_C1 [2]
Auth Id         : user1 [3]
 Total Segment :          1 [4] Max Sum Segment :          1 [6]
 Total Page    :          5 [5] Max Sum Page    :          5 [7]
 RD Area Count :          1/         1 [8]

--------------------------------------------------------------------------------
RD Area Name    : user_rdarea_1 [9]
 Server         : bes1 [10]
 Status         :   [22]
 Original RD Area Name : user_rdarea_1 [23]
 Generation Number :  0[24] Replica RD Area Count :  0[25]
 [26]     [27]              [28]              [29]
 History1 Hold Status :      Hold Code :     0  Hold Time :
 History2 Hold Status :      Hold Code :     0  Hold Time :
```

```
Unused Segment:         24 [11]
        Used(Full)       Used(      Full)        Sum
        [12]  [13]        [14]       [15]        [16]
Segment 100%(  0%)          1(         0)          1
        [17]  [18]        [19]       [20]        [21]
Page    100%(  0%)          5(         0)          5

Collect Prearranged Page :          0 [30]
Collect On Page :          0 [31]
Used Page Ratio       Page(Ratio) [32]
           0% :          0(   0%)
       1- 10% :          1(  20%)
      11- 20% :          0(   0%)
      21- 30% :          3(  60%)
      31- 40% :          0(   0%)
      41- 50% :          1(  20%)

      51- 60% :          0(   0%)
      61- 70% :          0(   0%)
      71- 80% :          0(   0%)
      81- 90% :          0(   0%)
      91-100% :          0(   0%)
Total                    5
```

Explanation

1. Date and time of completion of acquisition of the information required for this condition analysis, in the format *YYYY/MM/DD hh:mm:ss*:

   YYYY: Year. *MM*: Month. *DD*: Date. *hh*: Hour. *mm*: Minute. *ss*: Second.
2. Name of the index subject to analysis.
3. Authorization identifier of the index subject to analysis.
4. Total number of segments allocated to the index (number of used segments + number of unused segments).
5. Total number of pages in the segments allocated to the index (used segments + unused segments).
6. Maximum total number of segments allocated to the index per RDAREA.
7. Maximum total number of pages in the segments allocated to the index per RDAREA.
8. Number of RDAREAs analyzed successfully and number of RDAREAs containing the index storage pages.
9. Name of an RDAREA subject to analysis.
10. Name of the server managing the RDAREA.
11. Total number of unused segments in the RDAREA.

12. Percentage ratio of used segments:

    ↑ number of used segments/value of 16 ↑ × 100 (%)

13. Percentage ratio of full segments:

    ↑ number of full segments/value of 16 ↑ × 100 (%)

14. Number of used segments.
15. Number of full segments.
16. Total number of segments allocated to the corresponding index in the RDAREA (number of used segments).
17. Percentage ratio of used pages:

    ↑ number of used pages/value of 21 ↑ × 100 (%)

18. Percentage ratio of full pages:

    ↑ number of full pages/value of 21 ↑ × 100 (%)

19. Number of used pages.
20. Number of full pages.
21. Total number of pages in the segments that are allocated to the corresponding index in the RDAREA (number of used pages + number of unused pages).
22. Status of unfinished index during index creation (if the index has been completed, this field is blank):

    `UNFINISH_0`

    See *Figures 15-9* and *15-10*.

    `UNFINISH_1`

    See *Figures 15-11* and *15-12*.

23. Name of the original RDAREA.
24. Generation number of the corresponding RDAREA.
25. Number of replica RDAREAs.
26. Shutdown history of the corresponding RDAREA.

    `History1`: Shutdown information immediately before the shutdown was released (if the RDAREA is currently shut down, this field displays the current shutdown information)

    `History2`: Shutdown information immediately preceding `History1`

27. Shutdown type of the corresponding RDAREA. If there is no history, this field is

blank.

`CMD`: Command shutdown by HiRDB error detection

`FLT`: Error shutdown

28. Code indicating the cause of the shutdown of the corresponding RDAREA. For details about the shutdown cause code, see *15.3.1(2)(a) For an RDAREA other than a LOB RDAREA*.
29. Shutdown time of the corresponding RDAREA. If there is no history, this field is blank.
30. Number of pages that can be released by `pdreclaim` (number of used free pages whose usage ratio is 0%).
31. Number of pages that are under release processing because `pdreclaim` is currently executing or has terminated due to an error. If `pdreclaim` has terminated with an error, re-execute it (the value of this field becomes 0). The displayed page count is included in the percentage ratio of used pages (0%).
32. Number of used pages for each 10% range. The parentheses enclose the percentage ratio to all used pages. Because the fraction part for each ratio is rounded up, the sum of all values may exceed 100. Note the following:
    - For a page that stores 201 or more duplicate keys, the percentage ratio of `PCTFREE` specified during table definition is ignored. Therefore, data is not rearranged according to the `PCTFREE` value even during reorganization.
    - For an index to which the index key value with no lock is applied, the deletion key value is treated as free space. If all used pages are deletion key values, the usage ratio is 0%.
    - A full page may not always fall in the usage ratio range of 91-100%.
    - A free space created by update processing that shortens an existing row is not treated as free space.
    - If the ratio of used pages is less than (100 - free space ratio specified with `PCTFREE`), the storage efficiency may have decreased. You should consider executing `pdrorg`. For an area that stores branch rows (variable-length character string data with a length of 256 bytes or greater, data for repetition columns, and data for columns of abstract data type), executing `pdrorg` has no effect on the storage efficiency.

Note 1

Nos. 23-25 are displayed when HiRDB Staticizer Option is used. If a generation is specified for analysis and the specified replica RDAREA generation does not exist, the utility issues a warning message, in which case no results are displayed.

Note 2

Nos. 30-32 are displayed when the `-d` option is specified.

- Plug-in index

```
pddbst VV-RR(Object Option) ** Index Analysis **           2002/03/29 18:05:06 [1]
Index Name     : IDX05 [2]
Auth Id        : MANUAL [3]
 Total Segment :       9501[4] Max Sum Segment :       9501 [6]
 Total Page    :       9501[5] Max Sum Page    :       9501 [7]
 RD Area Count :          1/         1 [8]

-------------------------------------------------------------------------------
RD Area Name   : USER_RDLOB_10 [9]
 Server        : BES02 [10]
 Status        :     [15]
 Original RD Area Name : USER_RDLOB_10 [19]
 Generation Number :  0[20] Replica RD Area Count :  0 [21]

   [22]                [23]                [24]             [25]
 History1 Hold Status :      Hold Code :     0  Hold Time :
 History2 Hold Status :      Hold Code :     0  Hold Time :
 Unused Segment:        499(       499) [11]
         [12]       [13]       [14]
         Used       Used        Sum
 Segment 100%       9501       9501
                [16]                         [17]
 Segment Over  : N  Last Segment :       9501/     10000
                [18]
 Lobmap  Over  : N
```

Explanation

1. Date and time of completion of acquisition of the information required for this condition in the format *YYYY/MM/DD hh:mm:ss*.

   YYYY: Year. *MM*: Month. *DD*: Day. *hh*: Hour. *mm*: Minute. *ss*: Second
2. Name of the index subject to analysis.
3. Authorization identifier for the index subject to analysis.
4. Total number of segments allocated to the index (number of used segments + number of unused segments).
5. Total number of pages in the segments allocated to the index (used segments + unused segments).
6. Maximum total number of segments allocated to the index per RDAREA.
7. Maximum total number of pages allocated to the index per RDAREA.
8. Number of RDAREAs analyzed successfully, and number of RDAREAs containing the index storage pages.

9. Name of the RDAREA subject to analysis.
10. Name of the server managing the RDAREA.
11. Total number of unused segments in the RDAREA. The parentheses enclose the total number of unused segments in HiRDB files that are not in frozen update status in the corresponding RDAREA.
12. Percentage ratio of used segments:

    $\uparrow$ number of used segments/value of 16 $\uparrow \times$ 100 (%)
13. Number of used segments.
14. Total number of segments allocated to the corresponding indexes in the RDAREA (number of used segments).
15. Status of an unfinished index during index creation (if the index has been completed, this field is blank).

    `UNFINISH_0`

    See *Figures 15-9* and *15-10*.

    `UNFINISH_1`

    See *Figures 15-11* and *15-12*.

    `UNFINISH_2`

    The plug-in index is being updated on a delayed basis. When a UAP using the plug-in index delayed batch creation function is being executed, or after the UAP finished executing, a batch index creation process using the database reorganization utility will not be executed. For details about the plug-in index delayed batch creation function, see the *HiRDB Version 8 System Operation Guide*.
16. Whether or not order is lost in the LOB RDAREA.

    `Y`: Order is lost.

    `N`: Order is not lost.
17. Position information that indicates the last segment that is used in a *last position/total number of segments* format. If `Segment Over` is `Y`, this field always indicates the last segment.
18. Whether or not the LOB management entries are all used:

    `Y`: All are used.

    `N`: There are unused entries.
19. Name of the original RDAREA.

20. Generation number of the corresponding RDAREA.
21. Number of replica RDAREAs.
22. Shutdown history of the corresponding RDAREA.

    `History1`: Shutdown information immediately before the shutdown was released (if the RDAREA is currently shut down, this field displays the current shutdown information)

    `History2`: Shutdown information immediately preceding `History1`
23. Shutdown type of the corresponding RDAREA. If there is no history, this field is blank.

    `CMD`: Command shutdown by HiRDB error detection

    `FLT`: Error shutdown
24. Code indicating the cause of the shutdown of the corresponding RDAREA. For details about the shutdown cause code, see *15.3.1(2)(a) For an RDAREA other than a LOB RDAREA*.
25. Shutdown time of the corresponding RDAREA. If there is no history, this field is blank.

Note

Nos. 19-21 are displayed when HiRDB Staticizer Option is used. If a generation is specified for analysis and the specified replica RDAREA generation does not exist, the utility issues a warning message, in which case no results are displayed.

(3) Analyzing the analysis results

You can analyze the results of condition analysis by table or index in the same manner as for condition analysis by RDAREA (logical analysis).

If the storage rows are not distributed uniformly among RDAREAs, you are not making the best use of the HiRDB/Parallel Server's performance. In such a case, take appropriate measures, such as checking and revising key range partitioning or applying flexible hash partitioning or FIX hash partitioning (or changing the hash functions). The result should be to partition the storage uniformly among the RDAREAs.

If you have analyzed the status of an RDAREA the stores the LOB attribute of a plug-in-provided abstract data type or an RDAREA that stores a plug-in index, the percentage of used segments in the RDAREA depends on the method used by the plug-in to allocate segments.

15.3.4 Cluster key and clustering data page storage condition analysis

(1) Purpose

This analysis enables you to determine the degree of disorganization in the storage conditions for cluster keys and clustering data pages, so that you can decide whether or not the database needs to be reorganized.

(2) Analysis results

The following shows the results of cluster key and clustering data page storage condition analysis:

```
pddbst VV-RR(Object Option) ** Index Status Analysis **     2001/08/22 16:44:01 [1]
Index Name : (CLUSTER0000131198) [2]
Auth Id    : MANUAL [3]
Table Name : TBL06 [4]


***** Cluster Key Analysis *****
 RD Area Count :          1/        1 [5]

-------------------------------------------------------------------------------
RD Area Name : USER_RDAREA_4                    Server : BES01 [6]
 Original RD Area Name : USER_RDAREA_4 [24]
 Generation Number :  0[25] Replica RD Area Count :  0[26]
 Segment Size:          5 Pages   Page Size :       4096 [7]
 Index Level :          1 [8]
                      Total            Repetitional
 Key Count              100 [9]                  0 [10]
 Row Count              100 [11]                 0 [12]
             Total   Disordered
 Segment          0 [13]       0(  0.0%) [14]
 Page             0 [15]       0(  0.0%) [16]


***** Clustering Table Analysis *****
 RD Area Count :          1/        1 [17]
-------------------------------------------------------------------------------
RD Area Name : USER_RDAREA_4                    Server : BES01 [18]
 Original RD Area Name : USER_RDAREA_4
 Generation Number :  0   Replica RD Area Count :  0
 Segment Size:          5 Pages   Page Size :       4096 [19]
             Total   Disordered
 Segment          0 [20]       0(  0.0%) [21]
 Page             0 [22]       0(  0.0%) [23]
```

Explanation

1. Date and time of completion of acquisition of the information required for this condition analysis, in the format *YYYY/MM/DD hh:mm:ss*.

   YYYY: Year. *MM*: Month. *DD*: Date. *hh*: Hour. *mm*: Minute. *ss*: Second.

2. Index name of the cluster key subject to analysis.
3. Authorization identifier of the index owner of the cluster key subject to analysis.
4. Name of the table that defined the cluster key subject to analysis.
5. Number of RDAREAs that store the index in 2 above, and the number of those RDAREAs whose information could be analyzed normally:

   Number of RDAREAs whose information could be analyzed normally / number of RDAREAs that store the index
6. Name of an RDAREA that stores the analyzed index, and the name of the server managing that RDAREA.
7. Segment size (number of pages per segment) and page length (in bytes) of the RDAREA in 6.
8. Number of index levels of the index key of the RDAREA in 6.
9. Number of stored keys in the index of the RDAREA in 6 (displayed as an integer of up to 15 digits; a value in excess of 15 digits is displayed exponentially).
10. Number of keys stored in a duplicate key structure[1] in the index of the RDAREA in 6 (same display format as in 9).

    [1] Refers to the structure in which management information for the relevant keys in the index page is stored in multiple pages when the number of duplicate key values exceeds 200.
11. Number of rows stored in the index of the RDAREA in 6 (same display format as in 9).
12. Number of rows corresponding to 10 (same display format as in 9).
13. Number of times the storage position of segments in the index of the RDAREA in 6 changed.
14. In relation to 13, number of times the storage order was incorrect (reverse direction). Parentheses indicate the ratio of segments in incorrect storage order (14/13 $\times$ 100%).
15. Number of times the storage position of pages in the index of the RDAREA in 6 changed.
16. In relation to 15, number of times the storage order was incorrect (reverse direction). Parentheses indicate the ratio of segments in incorrect storage order (16/15 $\times$ 100%).
17. Number of RDAREAs stored in the table in 3, and the number of those RDAREAs whose information could be analyzed normally:

    Number of RDAREAs that could be analyzed normally / number of RDAREAs that store the table

18. Name of an RDAREA that stores the analyzed table, and the name of the server that manages that RDAREA.
19. Segment size (number of pages per segment) and page length (in bytes) of the RDAREA in 18.
20. Number of times the storage position of segments in the index of the RDAREA in 18 changed.
21. In relation to 20, number of times the storage order was incorrect (reverse direction). Parentheses indicate the ratio of segments in incorrect storage order (21/20 × 100%).
22. Number of times the storage position of table pages stored in the RDAREA in 18 changed.
23. In relation to 22, number of times the storage order was incorrect (reverse direction). Parentheses indicate the ratio of segments in incorrect storage order (23/22 × 100%).
24. Name of the original RDAREA.
25. Generation number of the corresponding RDAREA.
26. Number of replica RDAREAs.

Note 1

If the cluster key is partitioned and stored in multiple RDAREAs, Nos. 6-17 are displayed for each such RDAREA. The information is displayed in ascending order of the RDAREA names.

Note 2

If the table is partitioned and stored in multiple RDAREAs, Nos. 18-23 are displayed for each such RDAREA. The information is displayed in ascending order of the RDAREA names.

Note 3

Nos. 24-26 are displayed when HiRDB Staticizer Option is used. If a generation is specified for analysis and the specified replica RDAREA generation does not exist, the utility displays * as the RDAREA name and displays only the original RDAREA name, generation number, and server name.

If there is no replica RDAREA for either a table storage RDAREA or a corresponding cluster key index RDAREA, the utility displays only the RDAREA name, original RDAREA name, generation number, and server name for the existing replica RDAREA. For the nonexistent replica RDAREA, the utility displays * as the RDAREA name and displays only the original RDAREA name, generation number, and server name, and omits all subsequent information. If the specified generation of a replica RDAREA for a cluster key index storage RDAREA does not exist, the utility issues a warning message without displaying analysis results.

(3) Analyzing the analysis results

Table 15-4 describes the results of cluster key and clustering data page condition analysis and explains the actions to be taken.

Note that row deletion has no effect on the degree of irregularity in the clustering data conditions in the analysis results. If deletion is the principal table operation performed, you must take the following steps to determine whether or not the database needs to be reorganized:

1. Determine the number of used free pages (by table condition analysis)
2. Determine the free area in terms of the numbers of storage rows and used pages

For 2, the following shows the size of the invalid free area:

（*Current number of used pages in table*[1] - *number of table storage pages*[2]） × *page size* (bytes)

[1] Obtain the value from the results of condition analysis by table.

[2] To obtain the value, see the *HiRDB Version 8 Installation and Design Guide*.

Table 15-4: Results of cluster key and clustering data page condition analysis and actions to be taken

Target	Analysis result	Description and action
Cluster keys	There are too many rows for the number of storage keys in the index.	The key duplication ratio is high, affecting the performance of search operations that use the index. Check and, if necessary, revise the column structure in the index definition.
	There are keys that are stored in a duplicate key structure.[1]	Some keys have (or had) a high degree of duplication, affecting the performance of search operations that use the index. Check and, if necessary, revise the column structure in the index definition (if this condition existed in the past, you can eliminate the duplicate key structure by using `pdrorg` to reorganize the index).[2]
	Degree of irregularity is high in the storage order.	There is irregularity in the storage order, which is affecting search performance. Use `pdrorg` to reorganize the index. If there is irregularity also in the storage order of data pages, reorganizing the table will automatically reorganize the index.
Clustering data pages	Degree of irregularity is high in the storage order.	There is irregularity in the storage order of data pages, which is affecting the performance of access operations in the order of cluster key values. Use `pdrorg` to reorganize the table.
	The following condition is true: *Storage location changes count > storage location changes count* or *number of used segments in the results of condition analysis by table > number of pages* - 1	

[1] This is a structure for storing on multiple pages the management information for the applicable key on the index page when there are 201 or more duplicate key values. Once a duplicate key structure is employed, the information management pages will not be deleted even if the duplicates count drops below 201.

[2] Re-define the table with the cluster, excluding columns with a high degree of data duplication in index component columns.

15.4 Execution results of the facility for predicting reorganization time

15.4.1 Output format of the execution results

The execution results of the facility for predicting reorganization time depend on the values of the `-e` and `-m` options. Table 15-5 shows the relationship between the options and the output information.

Table 15-5: Relationship between the options and the output information (facility for predicting reorganization time)

-e option specification	-m option	Information about scheduled database maintenance date	Information about maintenance method	Information for each analysis item
`1` (prediction level 1)	Specified	Y	Y	—
	Omitted	Y	—	—
`2` (prediction level 2)	N/A	Y	Y	Y

Legend:

Y: The corresponding information is output.

— : The corresponding information is not output.

N/A: Cannot be specified.

(1) Scheduled database maintenance date

This example displays RDAREAs whose segment usage rate is predicted to exceed the standard value within the monitoring period. The RDAREAs are displayed in chronological order of the scheduled database maintenance dates.

```
 pddbst vv-rr       ***** Rdarea resource forecast *****     yyyy/mm/dd hh:mm:ss
No         Date        RdareaName                        ResKind
---------- ----------- -------------------------------- ----------
         1 2005/05/31  "RDUSER01"                        Segment
         2 2005/06/01  "RDUSER12"                        Segment
         3 2005/06/03  "rd0002"                          Segment
         4 2005/06/04  "lobrdarea701"                    Segment
         5 2005/06/11  "RDUSER02"                        Segment
---------- ----------- -------------------------------- ----------
```

Note

For a HiRDB/Parallel Server, a scheduled database maintenance date is based on the dictionary server's date and time settings.

Explanation

`No`:

Analysis result number (maximum of 10 digits)

`Date`:

Date predicted by the facility on which the segment usage rate will exceed the standard value (scheduled database maintenance date) (in the format *YYYY/MM/DD*)

`RdareaName`:

Name of the RDAREA whose maintenance is predicted to be necessary (maximum of 32 characters)

`ResKind`:

Item that is predicted to exceed its standard value (maximum of 10 characters):

`Segment`: Segment usage rate

(2) Maintenance method

This example displays RDAREAs whose segment usage rate is predicted to exceed the standard value within the monitoring period. The RDAREAs are displayed in chronological order of the scheduled database maintenance dates, together with the recommended maintenance method. In prediction level 2, of the tables and indexes for which maintenance is determined to be necessary from the information by analysis item, the example displays those that affect maintenance of RDAREAs.

```
 pddbst vv-rr       ***** Maintenance Information *****       yyyy/mm/dd hh:mm:ss
No          : 1
Rdarea Name : "RDUSER01"
Method      :
Segment     : 42560
-----------------------------------------------------------------------------------
--
 Reclaim      Reorganize   Type Name                                   Date
------------- ------------ ---- -------------------------------------- ----------
      28089 *       9363    T   "k1234567"."table01"                   2005/05/31
          0        12342 *  T   "k1234567"."table02"                   2005/05/31
        851         7235 *  I   "k1234567"."index01"                   2005/05/31
          3          -10    T   "k1234567"."table10"                   2005/05/31
==================================================================================
No          : 2
Rdarea Name : "RDUSER12"
Method      :
Segment     : 32985
------------------------------------------------------------------------------------
--
 Reclaim      Reorganize   Type Name                                     Date
------------- ------------ ---- ---------------------------------------- ----------
       3628        23749 *  T   "k1234567"."table01"                     2005/06/01
       1319         5937 *  T   "k1234567"."table03"                     2005/06/01
       1649 *       3628    I   "k1234567"."index01"                     2005/06/01
       2638 *        989    T   "k1234567"."table11"                     2005/06/01
==================================================================================

No          : 3
Rdarea Name : "rd0002"
Method      : Expand
Segment     : 1523
------------------------------------------------------------------------------------
--
 Reclaim      Reorganize   Type Name                                     Date
------------- ------------ ---- ---------------------------------------- ----------
        228         1035    T   "k1234567"."table01"                     2005/06/03
          0          121    T   "k1234567"."table03"                     2005/06/03
         30           60    I   "k1234567"."index07"                     2005/06/03
==================================================================================

No          : 4
Rdarea Name : "lobrdarea701"
Method      :
Segment     : 2498
------------------------------------------------------------------------------------
--
 Reclaim      Reorganize   Type Name                                     Date
------------- ------------ ---- ---------------------------------------- ----------
          0         2764 *  L   "k1234567"."table701"                    2005/06/04
==================================================================================

No          : 5
Rdarea Name : "RDUSER02"
Method      : Expand
Segment     : 31854
------------------------------------------------------------------------------------
--
 Reclaim      Reorganize   Type Name                                     Date
------------- ------------ ---- ---------------------------------------- ----------
       3503        23934    T   "k1234567"."table21"                     2005/06/11
       1592         3503    I   "k1234567"."index22"                     2005/06/11
       2548          955    T   "k1234567"."table23"                     2005/06/11
==================================================================================
```

Note

For a HiRDB/Parallel Server, a scheduled database maintenance date is based on the dictionary server's date and time settings.

Explanation

`No`:

Analysis result number (maximum of 10 digits)

`Rdarea Name`:

Name of the RDAREA whose maintenance is predicted to be necessary (maximum of 32 characters)

`Method`:

RDAREA maintenance method (maximum of 20 characters):

`Expand`:

Expand the RDAREA

`Extend`:

Maintenance is not necessary (HiRDB will perform automatic extension of the RDAREA)

`Reinit`:

Re-initialize the RDAREA

blank:

Maintenance is not necessary

`Segment`:

Number of segments to be released (maximum of 10 digits)

- In prediction level 1

  This is the minimum number of segments that need to be allocated during maintenance.

  If `Expand` is displayed for `Method`, expand the RDAREA so that the actual number of segments will be greater than the value displayed here.

- In prediction level 2

  This is the minimum number of segments that need to be allocated during maintenance. If maintenance of a table or index is predicted to be necessary, the displayed value reflects the result of executing such necessary maintenance. For example, if `Expand` is displayed for `Method`, the minimum number of segments that must be allocated is displayed, assuming that the necessary maintenance will be executed prior to RDAREA expansion.

`Reclaim`:

Number of segments predicted to be released when used free segments are released after used free pages are released (maximum of 13 digits)

If the facility predicts that release of used free segments is necessary as the table or index maintenance method, an asterisk (`*`) is displayed to the right of this item.

`Reorganize`:

Number of segments predicted to be released when reorganization is executed (maximum of 13 digits)

If the facility predicts that reorganization is necessary as the table or index maintenance method, an asterisk (`*`) is displayed to the right of this item.

If the facility predicts that reorganization will result in many data storage pages, a negative value (with a minus sign) is displayed.

`Type`:

Type of maintenance target (1 character):

`T`: Table

`I`: Index

`L`: LOB RDAREA

`Name`:

Name of the table or index (including authorization identifier) that is to be maintained (maximum of 52 characters). When `Type` is `L`, the facility displays the name of the table for which the LOB column has been defined.

The output format is `"`*authorization-identifier*`"."`*table-identifier*`"` or `"`*authorization-identifier*`"."`*index-identifier*`"`. For a data dictionary table, the authorization identifier is `"(Data dictionary)"`.

`Date`:

Scheduled database maintenance date (in the format *YYYY/MM/DD*)

- Supplement for prediction level 2

In prediction level 2 (`-e 2`), the output information may appear as described below. This information is output when maintenance of a table or index is determined to be necessary from the information by analysis item.

1. An asterisk (*) indicating a need for reorganization is displayed for a resource for which the number of used segments is increased as a result of reorganization.
2. Maintenance times may differ within the same RDAREA.
3. Both expansion and reorganization are indicated for an RDAREA.

The following shows an output example, where [1] through [3] correspond to the numbers above:

```
 pddbst VV-RR       ***** Maintenance Information *****       yyyy/mm/dd hh:mm:ss
No           : 1
Rdarea Name : "RDUSER01"
Method       :
Segment      : 42560
-----------------------------------------------------------------------------------
--
 Reclaim      Reorganize   Type Name                                         Date
------------- ------------ ---- -------------------------------------------
----------
          3          10 *  T   "k1234567"."table10"                  [2] 2005/05/16
      28089 *       9363    T   "k1234567"."table01"                      2005/05/31
          0       12342 *  T   "k1234567"."table02"                      2005/05/31
        851        7535 *  I   "k1234567"."index01"                      2005/05/31
          3      [1] -10 *  T   "k1234567"."table11"                      2005/05/31
          3     [1] -300 *  T   "k1234567"."table12"                      2005/05/31
===================================================================================
==

No           : 2
Rdarea Name : "rd0002"
Method       : Expand [3]
Segment      : 1523
-----------------------------------------------------------------------------------
--
 Reclaim      Reorganize   Type Name                                         Date
------------- ------------ ---- -------------------------------------------
----------
        228     [3] 1035 *  T   "k1234567"."table01"                      2005/06/03
          0      [3] 121 *  T   "k1234567"."table03"                      2005/06/03
         30       [3] 60 *  I   "k1234567"."index07"                      2005/06/03
          3     [3] -300 *  T   "k1234567"."table22"                      2005/06/03
===================================================================================
==
```

(3) Information by analysis item

This example displays in chronological order of the scheduled database maintenance dates the RDAREAs, tables, and indexes whose segment usage has exceeded a standard value within the monitoring interval. Also included is the maintenance method, as well as detailed information about the resources that are displayed for the maintenance method.

```
 pddbst VV-RR       ***** Analysis Item Data *****       yyyy/mm/dd hh:mm:ss
No        : 1
Date      : 2005/05/25
Method    : Reclaim
Type      : T
Name      : "k1234567"."table01" ("RDUSER01")
StateDate : 2005/05/19 00:10:32

-------------------------------------------------------------------------------------
Information                               Value      Method               Date
---------------------------------------- ----------- -------------------- ----------
Empty Page Ratio                          40 ( 50) * Reclaim pages        2005/05/25
Unused Page Ratio (30)                    20 ( 30)
Used Segment for Cluster                  40 ( 50)
Unused Page Differ From PCTFREE (10,10)   50 ( 75)
=====================================================================================
No        : 2
Date      : 2005/05/31
Method    :
Type      : R
Name      : "RDUSER01"
StateDate : 2005/05/19 00:10:32

-------------------------------------------------------------------------------------
Information                               Value      Method               Date
---------------------------------------- ----------- -------------------- ----------
Used Segment Ratio (0,10)                 40 ( 80)                        2005/05/31
=====================================================================================
No        : 3
Date      : 2005/06/01
Method    : Reorganize
Type      : T*
Name      : "k1234567"."table10" ("RDUSER01")
StateDate : 2005/05/19 00:10:32

-------------------------------------------------------------------------------------
Information                                Value     Method               Date
---------------------------------------- ----------- -------------------- ----------
Empty Page Ratio                          25 ( 50)   Reclaim pages        2005/08/10
Unused Page Ratio (30)                    25 ( 30) * Reorganize           2005/06/01
Used Segment for Cluster                  40 ( 50)
Unused Page Differ From PCTFREE (10,10)   60 ( 75)
=====================================================================================
No        : 4
Date      : 2005/06/03
Method    : Expand
Type      : R
Name      : "rd0002"
StateDate : 2005/05/19 00:10:32
```

```
-----------------------------------------------------------------------------------
Information                                Value      Method               Date
---------------------------------------- ----------- -------------------- ----------
Used Segment Ratio (0,10)                  78 ( 95)   Expand               2005/06/03
===================================================================================
No        : 5
Date      : 2005/06/04
Method    : Reclaim
Type      : I
Name      : "k1234567"."index07" ("rd0002")
StateDate : 2005/05/19 00:10:32

-----------------------------------------------------------------------------------
Information                                 Value     Method               Date
---------------------------------------- ----------- -------------------- ----------
Empty Page Ratio                           38 ( 50)   Reclaim              2005/06/04
Unused Page Ratio (30)                     20 ( 30)
Unused Page Differ From PCTFREE (10,10)    70 ( 75)
===================================================================================
No        : 6
Date      : 2005/06/04
Method    : Reorganize
Type      : L
Name      : "k77a777"."table707" ("lobrdarea701")
StateDate : 2005/05/19 00:10:32

-----------------------------------------------------------------------------------
Information                                 Value     Method               Date
---------------------------------------- ----------- -------------------- ----------
Used Segment for LOB Columns (0,10)        70 ( 80) * Reorganize           2005/06/04
===================================================================================
No        : 7
Date      : 2005/06/06
Method    : Expand
Type      : R
Name      : "RDUSER12"
StateDate : 2005/05/19 00:10:32

-----------------------------------------------------------------------------------
Information                                 Value     Method               Date
---------------------------------------- ----------- -------------------- ----------
Used Segment Ratio (0,10)                  70 ( 80) * Expand               2005/07/01
===================================================================================
:
```

Note

In the case of a HiRDB/Parallel Server, a scheduled database maintenance date is based on the dictionary server's date and time settings.

Explanation

`No`:

Analysis result number (maximum of 10 digits). This corresponds to the number assigned to the information about each scheduled database maintenance date.

`Method`:

Maintenance method for the target type (`Type`) (maximum of 14 characters):

`ReclaimS`

Use `pdreclaim` to release used free segments.

`ReclaimP`

Use `pdreclaim` to release used free pages.

`Reorganize`

Use `pdrorg` to perform reorganization. When reorganization is executed, the area to be used may increase because free area is allocated according to the specified `PCTFREE` value. If there is a shortage of space, extend the RDAREA beforehand.

`Expand`

Use `pdmod` to expand the RDAREA.

`Extend`

Maintenance is not needed (HiRDB performs automatic extension of the RDAREA). If HiRDB cannot allocate the area required for automatic extension, the user must use `pdmod` to extend the RDAREA.

`Reinit`

Re-initialize the RDAREA. If 16 HiRDB files have already been allocated to the RDAREA, the RDAREA can no longer be extended. In such a case, you must take appropriate action, such as re-initializing the RDAREA to increase the size and number of HiRDB files.

blank

No maintenance is needed.

`Type`:

Target type for which maintenance is determined to be necessary (2 characters):

`T`: Table

`I`: Index

`L`: LOB RDAREA

`R`: RDAREA

An asterisk (*) indicates that the target type is displayed in the information by analysis type, but not in the information about the maintenance method (example: `T*`). Therefore, in the case of a target type with an asterisk displayed, you must perform the indicated maintenance. In the case of a target type without an asterisk displayed, perform only the maintenance indicated in the information about the maintenance method.

`Name`:

Name of the RDAREA, table, or index (including the authorization identifier) that is the target of maintenance (maximum of 78 characters)

For an RDAREA

When RDAREA extension is instructed: "*RDAREA-name*"

For a LOB RDAREA:
"*authorization-identifier*"."*table-identifier*"("*RDAREA-name*")

For a table

For a non-row-partitioned table:
"*authorization-identifier*"."*table-identifier*"

For a row-partitioned table:
"*authorization-identifier*"."*table-identifier*"("*RDAREA-name*")

For an index

For a non-row-partitioned table:
"*authorization-identifier*"."*index-identifier*"

For a row-partitioned table:
"*authorization-identifier*"."*index-identifier*"("*RDAREA-name*")

`StateDate`:

Date and time the most recent information was acquired in the information used for analysis (in the format *YYYY*/*MM*/*DD* *hh*:*mm*:*ss*)

`Information`:

Names of analyzed items and the standard values used for the analyses. For details about the item names and standard values, see *15.5 Standard value definition file (facility for predicting reorganization time)*.

`Value`:

Analysis values. Each set of analysis values is output in the format *analysis-value* (*standard-value*). For details about the standard values, see *15.5 Standard value definition file (facility for predicting reorganization time)*.

An asterisk (*) is displayed following the set of analysis values for the item with the earliest prediction date among all the analyzed items for which maintenance was determined to be necessary as a result of prediction processing.

`Date`:

Scheduled database maintenance date (in the format *YYYY/MM/DD*).

15.4.2 Output format in CSV format

If you execute the facility for predicting reorganization time with the `predict` statement specified, you can output the prediction results to a file in CSV format.

(1) Rules for output data

1. The items are separated by the comma (`,`).
2. Each line of data ends with the linefeed character (`0x0a`).
3. If there is no information to be output for an item, only the comma is output (or the linefeed character in the case of the last item).
4. A title line is output as the first line.

(2) Prediction results output items

Table 15-6 describes the execution results items of the facility for predicting reorganization time that are output in the CSV format.

Table 15-6: Execution results items of the facility for predicting reorganization time (output in CSV format)

Title	Output item	Output format	Max. length (bytes)	Output information		
				Scheduled database maintenance date	Maintenance method	Information by analysis item
`kind`	Output type: `p`: Information about scheduled database maintenance date `m`: Information about maintenance method `d`: Information by analysis item	Character string	8	Y	Y	Y
`No`	Analysis result number	Numeric value	10	Y	Y	Y

Title	Output item	Output format	Max. length (bytes)	Output information		
				Scheduled database maintenance date	Maintenance method	Information by analysis item
`Date`	Scheduled database maintenance date	*YYYY/ MM/DD*	10	Y	Y	Y
`Type`	Target type: `T`: Table `I`: Index `L`: LOB RDAREA `R`: Data dictionary RDAREA, user RDAREA, or registry RDAREA	Character string	1	Y	Y	Y
`*`	Asterisk identifies a resource whose maintenance has been determined to be necessary. These are the resources the user must maintain.	Character	1	Space	Y	Space
`AuthID`	Authorization identifier For a data dictionary table, `(Data dictionary)` is displayed.	Character string	30	N	Y*	Y*
`Name`	Name of the table or index	Character string	30	N	Y*	Y*

Title	Output item	Output format	Max. length (bytes)	Output information		
				Scheduled database maintenance date	Maintenance method	Information by analysis item
`Rdarea`	RDAREA name	Character string	30	Y	Y	Y
`Method`	Maintenance method: `0`: Maintenance is not necessary `1`: `ReclaimS` (release used free segments) `2`: `ReclaimP` (release used free pages) `3`: `Reorganize` (reorganize) `4`: `Expand` (expand RDAREA) `5`: `Extend` (extend RDAREA automatically (maintenance not necessary)) `6`: `Reinit` (Re-initialize RDAREA) For details about the maintenance method, see *15.4.1 Output format of the execution results*.	Numeric value	5	N	Y	Y

Title	Output item	Output format	Max. length (bytes)	Output information		
				Scheduled database maintenance date	Maintenance method	Information by analysis item
`NextExec`	Recommended time for the next execution of the condition analysis result accumulation facility	*YYYY/MM/DD*	10	N	N	N
`StateDate`	Analysis information acquisition date and time. This is the date and time the most recent information was acquired among all analysis information that was used for the analysis.	*YYYY/MM/DD hh:mm:ss*	19	Y	N	Y

Title	Output item	Output format	Max. length (bytes)	Output information		
				Scheduled database maintenance date	Maintenance method	Information by analysis item
`InfoNo`	Number of the analysis item type: `1`: Empty Page Ratio `2`: Unused Page Ratio `3`: Number of Branch Row `8`: Used Segment for LOB Columns `10`: Used Segment for Cluster `11`: Unused Page Differ From PCTFREE `13`: Used Segment Ratio	Numeric value	5	Y	N	Y
`Value`	Analysis value for the analysis item type	Numeric value	5	Y	N	Y
`PredictBase`	Standard value used for the analysis item type	Numeric value	5	Y	N	Y
`Count`	Number of accumulated data items	Numeric value	5	N	N	N
`Segment`	Number of segments released	Numeric value	10	N	Y	N

Title	Output item	Output format	Max. length (bytes)	Output information		
				Scheduled database maintenance date	Maintenance method	Information by analysis item
`Reclaim`	Predicted number of released segments in releasing used free segments	Numeric value	10	N	Y*	N
`Reorganize`	Predicted number of released segments in reorganizatio n	Numeric value	10	N	Y*	N
`ItemMethod`	Number of the maintenance method for each analysis item	Numeric value	5	N	Y	Y
`CheckNo`	Number of standard values for checking	Numeric value	5	Y	N	Y
`Check1`	Standard value for checking by analysis item type used for analysis	Numeric value	5	Y	N	Y
`Check2`	Standard value for checking by analysis item type used for analysis	Numeric value	5	Y	N	Y

Legend:

Y: Output

N: Not output

blank: 1 space character is output

Note

An item is enclosed in double quotation marks (") if its output format is character string. The maximum length does not include these quotation marks.

* This information is not output when the target type is R.

(3) Output example

The following shows an output example in the CSV format:

(a) When the -e 1 option is specified and the -m option is omitted

```
Kind,No,Date,Type,*,AuthID,Name,Rdarea,Method,NextExec,StateDate,InfoNo,Value,Predic
tBase,
Count,Segment,Reclaim,Reorganize,ItemMethod,CheckNo,Check1,Check2
"p",1,2005/01/26,"R"," ",,,"RDUSER12",,,2004/12/27 15:58:08,13,50,80,,,,,,2,0,0
"p",1,2005/01/30,"R"," ",,,"rd0002",,,2004/12/27 15:58:08,13,60,80,,,,,,2,0,0
"p",1,2005/01/11,"R"," ",,,"lobrdarea701",,,2004/12/27 15:58:08,13,70,80,,,,,,2,0,0
"p",1,2005/01/15,"R"," ",,,"RDUSER02",,,2004/12/27 15:58:08,13,65,80,,,,,,2,0,0
```

(b) When the -e 1 and -m options are both specified

```
Kind,No,Date,Type,*,AuthID,Name,Rdarea,Method,NextExec,StateDate,InfoNo,Value,Predic
tBase,
Count,Segment,Reclaim,Reorganize,ItemMethod,CheckNo,Check1,Check2
"p",1,2005/01/26,"R"," ",,,"RDUSER01",,,2004/12/27 15:58:08,13,50,80,,,,,,2,0,0
"m",1,2005/01/26,"R"," ",,,"RDUSER01",0,,,,,,,2,0,0,0,,,
"m",1,2005/01/26,"I"," ","k1234567","index01","RDUSER01",0,,,,,,,2,0,1,0,,,
"m",1,2005/01/26,"T","*","k1234567","table01","RDUSER01",3,,,,,,,2,0,4,3,,,
          :
"p",1,2005/01/30,"R"," ",,,"RDUSER02",,,2004/12/27 15:58:08,13,50,80,,,,,,2,0,0
"m",1,2005/01/30,"R"," ",,,"RDUSER02",0,,,,,,,0,0,0,0,,,
"m",1,2005/01/30,"T"," ","k1234567","table03","RDUSER02",0,,,,,,,0,0,0,0,,,
          :
"p",1,2005/01/21,"R"," ",,,"RDUSER02",,,2004/12/27 15:58:08,13,50,80,,,,,,2,0,0
"m",1,2005/01/21,"R","*",,,"RDUSER02",4,,,,,,,6,0,0,4,,,
"m",1,2005/01/21,"I"," ","k1234567","index03","RDUSER02",0,,,,,,,6,0,1,0,,,
"m",1,2005/01/21,"T"," ","k1234567","table03","RDUSER02",0,,,,,,,6,0,4,0,,,
          :
```

(c) When the -e 2 option is specified

```
Kind,No,Date,Type,*,AuthID,Name,Rdarea,Method,NextExec,StateDate,InfoNo,Value,Predic
tBase,CheckNo,Check1,Check2
"p",1,2005/05/31,"R",,,,"RDUSER01",,2005/05/28,2005/04/12
17:30:12:42,13,78,80,2,0,10
"m",1,2005/05/31,"T","*","k1234567","table01","RDUSER01",1,,,,,,0,,
"m",1,2005/05/31,"T",,"k1234567","table02","RDUSER01",2,,,,,,0,,
"m",1,2005/05/31,"I","*","k1234567","index01","RDUSER01",2,,,,,,0,,
"m",1,2005/05/31,"T","*","k1234567","table10","RDUSER01",2,,,,,,0,,
"d",1,2005/05/31,"T",,"k1234567","table01","RDUSER01",1,2005/05/28,2005/04/12
17:30:12:42,1,40,50,0,,
"d",1,2005/08/10,"T","*","k1234567","table10","RDUSER01",1,2005/07/31,2005/04/12
17:30:12:42,1,25,50,0,,

"d",1,2005/06/01,"T","*","k1234567","table10","RDUSER01",2,2005/05/29,2005/04/12
17:30:12:42,2,25,30,1,50,
"p",2,2005/06/01,"R",,,,"RDUSER12",,2005/05/29,2005/04/12
17:30:12:42,13,78,80,2,0,10
"m",2,2005/06/01,"T","*","k1234567","table01","RDUSER12",2,,,,,,0,,
"m",2,2005/06/01,"T","*","k1234567","table03","RDUSER12",2,,,,,,0,,
"m",2,2005/06/01,"I","*","k1234567","index01","RDUSER12",1,,,,,,0,,
"m",2,2005/06/01,"T","*","k1234567","table11","RDUSER12",1,,,,,,0,,
"p",3,2005/06/03,"R",,,,"rd0002",,2005/05/30,2005/04/12 17:30:12:42,13,78,80,2,0,10
"m",3,2005/06/03,"R",,,,"rd0002",3,,,,,,0,,
"d",3,2005/06/03,"R",,,,"rd0002",3,2005/05/30,2005/04/12 17:30:12:42,13,78,80,2,0,10
          :
```

15.5 Standard value definition file (facility for predicting reorganization time)

The facility for predicting reorganization time makes its predictions on the basis of predefined standard values for items. You can change these standard values as appropriate to your environment. You can specify in the standard value definition file the new standard values you wish to use for any items.

(1) Format

```
<threshold>
item-name=standard-value
      :
[rdarea=RDAREA-name]
item-name=standard-value
      :
[table=authorization-identifier.table-identifier]
item-name=standard-value
      :
[index=authorization-identifier.index-identifier]
item-name=standard-value
      :
```

(2) Explanation

`<threshold>`

Specifies that standard values for common items follow. When `<threshold>` is specified, it must be specified as the first line of the file.

`[rdarea=`*RDAREA-name*`]`

Specifies an RDAREA for which standard values are to be changed.

`[table=`*authorization-identifier*`.`*table-identifier*`]`

Specifies the table(s) for which standard values are to be changed.

To include all tables under the specified authorization identifier, specify `all` for *table-identifier*. To specify the data dictionary table, omit the authorization identifier.

`[index=`*authorization-identifier*`.`*index-identifier*`]`

Specifies the index(es) for which standard values are to be changed.

To include all indexes under the specified authorization identifier, specify `all` for *index-identifier*.

item-name=standard-value

Specifies the name of an item whose standard value is to be changed and the new standard value. For details about the item names that can be specified, see *(4) Names of items whose standard value can be changed*.

Rules

1. If the same item name is specified more than once, the last specification takes effect.
2. An item whose value is set to the asterisk (`*`) will be excluded from the checking performed by the facility for predicting reorganization time. There is no need to reorganize a table on which `PURGE TABLE` and initial data loading (`pdload` with existing data deleted) are executed periodically. To remove these resources as targets of the facility for predicting reorganization time, specify an asterisk (*). The facility for predicting reorganization time performs prediction processing assuming that no periodic maintenance such as `PURGE TABLE` is performed.

(3) Specification rules for the standard value definition file

1. An item specified before the first `[rdarea=`*RDAREA-name*`]`, `[table=`*authorization-identifier*`.`*table-identifier*`]`, or `[index=`*authorization-identifier*`.`*index-identifier*`]` is treated as a common item.
2. If the same `<threshold>`, `[rdarea=`*RDAREA-name*`]`, `[table=`*authorization-identifier*`.`*table-identifier*`]`, or `[index=`*authorization-identifier*`.`*index-identifier*`]` is specified more than once, the last specification takes effect.
3. If the same `[rdarea=`*RDAREA-name*`]`, `[table=`*authorization-identifier*`.`*table-identifier*`]`, or `[index=`*authorization-identifier*`.`*index-identifier*`]` containing the same item name is specified more than once, the last specification takes effect.
4. Each of <threshold>, `[rdarea=`*RDAREA-name*`]`, `[table=`*authorization-identifier*`.`*table-identifier*`]`, `[index=`*authorization-identifier*`.`*index-identifier*`]`, and *item-name=standard-value* must be specified on a separate line.
5. There can be no spaces or tab characters within *item-name=standard-value* or before *item-name*.
6. The utility ignores any information enclosed between `/*` and `*/`. The utility also ignores null lines.
7. If the standard value definition file contains an error, the utility terminates with an error.

8. If an RDAREA name, authorization identifier, table identifier, or index identifier contains a lowercase alphabetic letter or space, you must enclose the entire name or identifier in double quotation marks ("). If one of these names or identifiers is not enclosed in double quotation marks ("), any lowercase alphabetic letters it includes are handled as uppercase letters.
9. Place a specification common to all tables or all indexes (`all` specified for the table identifier or index identifier) before specifications for specific tables, indexes, or RDAREAs.

(4) Names of items whose standard value can be changed

Table 15-7 describes the items whose standard value can be changed. Table 15-8 provides guidelines for changing standard values.

Table 15-7: Items whose standard value can be changed

No.	Item name	Specifiable values	Standard value	Description	Whether or not changeable		
					R	T	I
1	`EMPTY_PAGE_RATIO`	`*`, 10-100	`30`	Among the total number of pages allocated, specifies the percentage that can be released by `pdreclaim` (%).	Y	Y	Y
2	`UNDER_PAGE_RATIO`	10-100	`10`, or `80-PCTFREE`	Specifies the page usage percentage to be used to check No. 3 (%).	Y	Y	Y
3	`UNUSED_PAGE_RATIO`	`*`, 10-100	`50`	Among the total number of pages allocated, specifies the percentage whose page usage is less than the value of No. 2 (%).	Y	Y	Y
4	`BRANCH_ROW`	`*`, 10-100	`50`	Among the total number of rows, specifies the percentage stored on multiple pages (%).	Y	Y	N
5	`USED_SEGMENT_LOB`	`*`, 10-100	`80`	Among the total number of segments, specifies the percentage derived from the last segment number in the LOB RDAREA (%).	Y	N	N

No.	Item name	Specifiable values	Standard value	Description	Whether or not changeable		
					R	T	I
6	USED_SEGMENT_RATIO_CLUS	*, 10-100	50	Among the total number of segments allocated to a clustering data page, specifies the percentage whose usage is 100% (%).	Y	Y	N
7	DIFF_PCTFREE_M	*, 10-100	10	Specifies the value used to determine the minimum page usage (%). This value is obtained by subtracting the minimum page usage rate from the page usage rate that is obtained based on the percentage of unused area specified in PCTFREE. If the specified value is greater than (100 - percentage of unused area specified in PCTFREE), the value of (100 - percentage of unused area specified in PCTFREE) is assumed as this value. If * is specified, only the value specified in No. 8 takes effect.	Y	Y	Y

No.	Item name	Specifiable values	Standard value	Description	Whether or not changeable		
					R	T	I
8	`DIFF_PCTFREE_P`	`*`, 10-100	`10`	Specifies the value used to determine the maximum page usage (%). This value is obtained by subtracting the page usage that is obtained based on the percentage of unused area specified in `PCTFREE` from the maximum page usage rate. If the specified value is greater than the percentage of unused area specified in `PCTFREE`, the percentage of unused area specified in `PCTFREE` is assumed as this value. If `*` is specified, only the value specified in No. 7 takes effect.	Y	Y	Y
9	`DIFF_PCTFREE_RATIO`	`*`, 10-100	`*`	Among the total number of pages allocated, specifies the percentage whose usage rate is less than the minimum value of No. 7 and greater than the maximum value of No. 8 (%). When this item is omitted, analysis is not performed for this item.	Y	Y	Y
10	`USED_SEGMENT`	`*`, 10-100	`80`	Among the total number of segments, specifies the percentage used throughout the entire RDAREA (%).	Y	N	N

No.	Item name	Specifiable values	Standard value	Description	Whether or not changeable		
					R	T	I
11	`EXTEND_COUNT_MIN`	0-24	`0`	Specifies the minimum number of times the RDAREA can be extended automatically. This value must satisfy the condition `EXTEND_COUNT_MIN` ≤ `EXTEND_COUNT`.	Y	N	N
12	`EXTEND_COUNT`	0-24	`0`	Specifies the number of times the RDAREA can be extended automatically. This value must satisfy the condition `EXTEND_COUNT_MIN` ≤ `EXTEND_COUNT`. `pddbst` cannot determine whether or not the HiRDB file system area becomes full as a result of automatic extension. Therefore, you must determine the value of `EXTEND_COUNT` taking into account the number of RDAREAs in the HiRDB file system area and the maximum number of extensions.	Y	N	N

Legend:

R: RDAREA

T: Table

I: Index

Y: Standard value can be changed.

N: Standard value cannot be changed.

Table 15-8: Guidelines for changing the standard value

No.	Item name	Standard value	How to determine whether or not maintenance is needed	Guideline for changing standard value
1	`EMPTY_PAGE_RATIO`	(*p*)=30%	If the percentage of used pages with a page usage rate of 0% is (*p*)% or greater, the utility determines that free pages must be released.	If there is not much space available in the RDAREA and you want to release invalid pages as soon as possible, reduce the value of (*p*).
2	`UNDER_PAGE_RATIO`	(*a*)=X-20%	If the percentage of used pages whose page usage rate of (*a*)% or greater is equal to or greater than (*p*)%, the facility determines that reorganization is necessary.	• If the storage row is extremely short for the page size and the page usage rate is less than (*a*)% immediately after the table is reorganized, reduce the value of (*a*). • If you want to reduce the percentage of pages with invalid area as much as possible, reduce the value of (*p*). • In the case of a table for which `NO SPLIT` is not specified and which contains variable-length character strings with a defined length of 256 bytes or greater, if more than 50% of the pages obtained immediately after reorganization have a page usage rate of (*a*)% or less (there is more branched variable-length character string data than non-branched data), increase the value of (*p*). • If there is not much space available in the RDAREA and you want to release the pages with invalid area as soon as possible, reduce the value of (*p*).
3	`UNUSED_PAGE_RATIO`	(*p*)=50%		

No.	Item name	Standard value	How to determine whether or not maintenance is needed	Guideline for changing standard value
4	`BRANCH_ROW`	(*p*)=50%	If the percentage of branched rows among the total number of rows is (*p*)% or greater, the facility determines that reorganization is necessary.	If there are many `UPDATE` processes that result in longer data and search performance needs to be maintained, reduce the value of (*p*).
5	`USED_SEGMENT_LOB`	(*p*)=80%	If the last segment number in the LOB RDAREA exceeds (*p*)% among the total number of segments and reorganization does not improve the condition, the utility determines that extension of the RDAREA is necessary. If reorganization results in a value less than (*p*), the facility determines that reorganization is necessary.	If the amount of data to be stored may increase suddenly and you want to take action at an early stage, reduce the value of (*p*).
6	`USED_SEGMENT_RATIO _CLUS`	(*p*)=50%	If the number of segments with a page usage per segment of 100% (segments without free pages) is equal to or greater than (*p*)% of the total number of segments in use, use of free space based on the percentage of unused pages, not the use of free space based on the percentage of unused area, has already begun; therefore, the facility determines that reorganization is necessary.	While data can be stored in the same page, clustering effects are still maintained. If you can determine from the storage status in the same segment that clustering is no longer effective, reduce the value of (*p*).

No.	Item name	Standard value	How to determine whether or not maintenance is needed	Guideline for changing standard value
7	`DIFF_PCTFREE_M`	(*a*)=X-10%	If the percentage of used pages with a page usage outside the range from (*a*)% to (*b*)% is equal to or greater than (*p*)%, the facility determines that reorganization is necessary.	If you want to perform reorganization even when the page usage rate is high (for a table in which `UPDATE`s resulting in a longer row occur frequently, or for a table with a cluster key defined in which there are frequent occurrences of `INSERT`s that involves data with an intermediate key), specify (*p*). In the case of an index with many duplicate key values, `PCTFREE` is ignored for the pages that store duplicate key values; therefore, if you specify (*p*), specify a sufficient value.
8	`DIFF_PCTFREE_P`	(*b*)=X+10%		
9	`DIFF_PCTFREE_RATIO`	(*p*)=*		
10	`USED_SEGMENT`	(*p*)=80%	If the number of used segments in the RDAREA is equal to or greater than (*p*)%, the facility determines that reorganization is necessary. If the value remains equal to or greater than (*p*)% after reorganization, the utility determines that extension is necessary. However, in the case of an RDAREA for which automatic extension has been specified, the utility does not recommend extension because the RDAREA can be extended automatically only up to (*a*) times.	• If you want to be able to detect a space shortage at an early stage when RDAREA extension is recommended, reduce the value of (*p*) (for example, if a space shortage occurs when there is no spare disk, you must first allocate a disk). • If the amount of data to be stored may increase suddenly and you want to take action at an early stage, reduce the value of (*p*).
11	`EXTEND_COUNT_MIN`	(*a*)=23		
12	`EXTEND_COUNT`			

Legend:

X: 100 - value of first argument of `PCTFREE`

(5) Standard value definition file specification example

The following shows a standard value definition file specification example:

```
<threshold>  .........................Common specification
USED_SEGMENT=70
USED_SEGMENT_LOB=65
  :
[table=all]  .........................Specification common to all tables
EMPTY_PAGE_RATIO=30
USED_SEGMENT=20
  :
[index=all]  .........................Specification common to all indexes
EMPTY_PAGE_RATIO=20
USED_SEGMENT=20
  :
[table=authorization-identifier.table-name]  ....Specification common to a specified table
EMPTY_PAGE_RATIO=35
USED_SEGMENT=30
  :
[index=authorization-identifier.index-name]  ...Specification common to a specified index
EMPTY_PAGE_RATIO=80
USED_PAGE_RATIO=60
  :
[rdarea=RDAREA-name]  ................Specification for an individual RDAREA
EXTEND_COUNT=10
  :
[table=authorization-identifier.table-name]  ...Specification for an individual table in an
RDAREA
EMPTY_PAGE_RATIO=35
USED_SEGMENT=30
  :
[index=authorization-identifier.index-name]  ...Specification for an individual index in an
RDAREA
EMPTY_PAGE_RATIO=80
USED_PAGE_RATIO=60
  :
```

15.6 Executing the condition analysis result accumulation facility in prediction level 2

The condition analysis result accumulation facility in prediction level 1 analyzes only directory pages because it acquires information only by RDAREA. For this reason, its execution time is comparatively brief.

In prediction level 2, the condition analysis result accumulation facility also acquires information by table and index. This requires analysis of data pages and index pages, resulting in a longer execution time than with prediction level 1.

If you execute the condition analysis result accumulation facility in prediction level 2 during online operations, online jobs may be affected. To minimize the effects on online jobs, you can choose one (or both) of the following execution methods:

- Interval analysis
- Merge analysis

The following subsection provides criteria for using these methods and describes how to execute them.

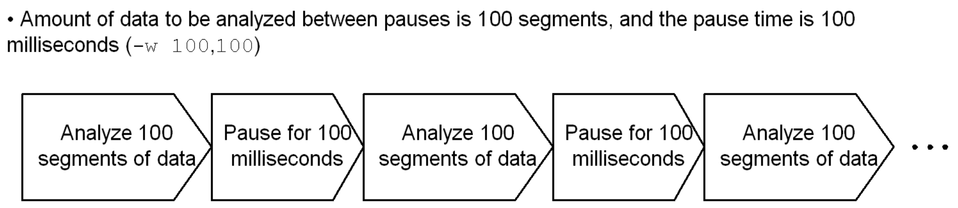
15.6.1 Interval analysis

Interval analysis sets pauses during execution of the condition analysis result accumulation facility. After analyzing a specified amount of data, the facility pauses the analysis for a specified amount of time. When the specified amount of time has elapsed, the facility analyzes the next set of the specified amount of data, then pauses again, repeating this cycle until all the data has been analyzed. Because the facility pauses, the adverse effects on online applications are reduced.

To execute interval analysis, you specify in the `-w` option the *amount of data to be analyzed (in segments)* between pauses and the *pause time (in milliseconds)*.

Figure 15-13 provides an overview of interval analysis.

Figure 15-13: Overview of interval analysis


• Amount of data to be analyzed between pauses is 100 segments, and the pause time is 100 milliseconds (-w 100,100)
Analyze 100 segments of data
Pause for 100 milliseconds
Analyze 100 segments of data
Pause for 100 milliseconds
Analyze 100 segments of data
...

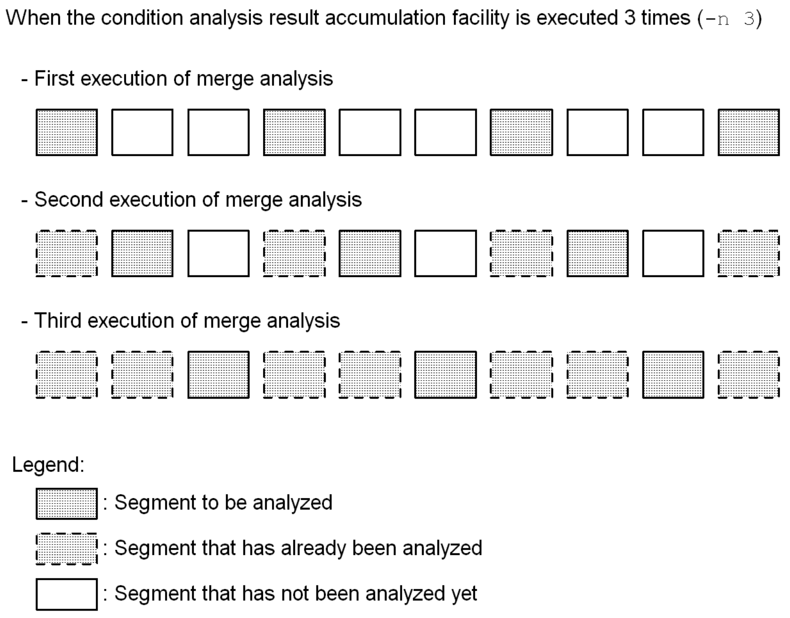
15.6.2 Merge analysis

Merge analysis executes the condition analysis result accumulation facility multiple times (over multiple days), accumulating and merging the condition analysis results. For example, it is effective to execute the facility over multiple days during off-peak hours when there are comparatively fewer online jobs.

To execute merge analysis, you specify in the `-n` option the number of times the facility is to be executed.

Figure 15-14 provides an overview of merge analysis. This figure assumes that a single RDAREA (10 segments) is to be analyzed.

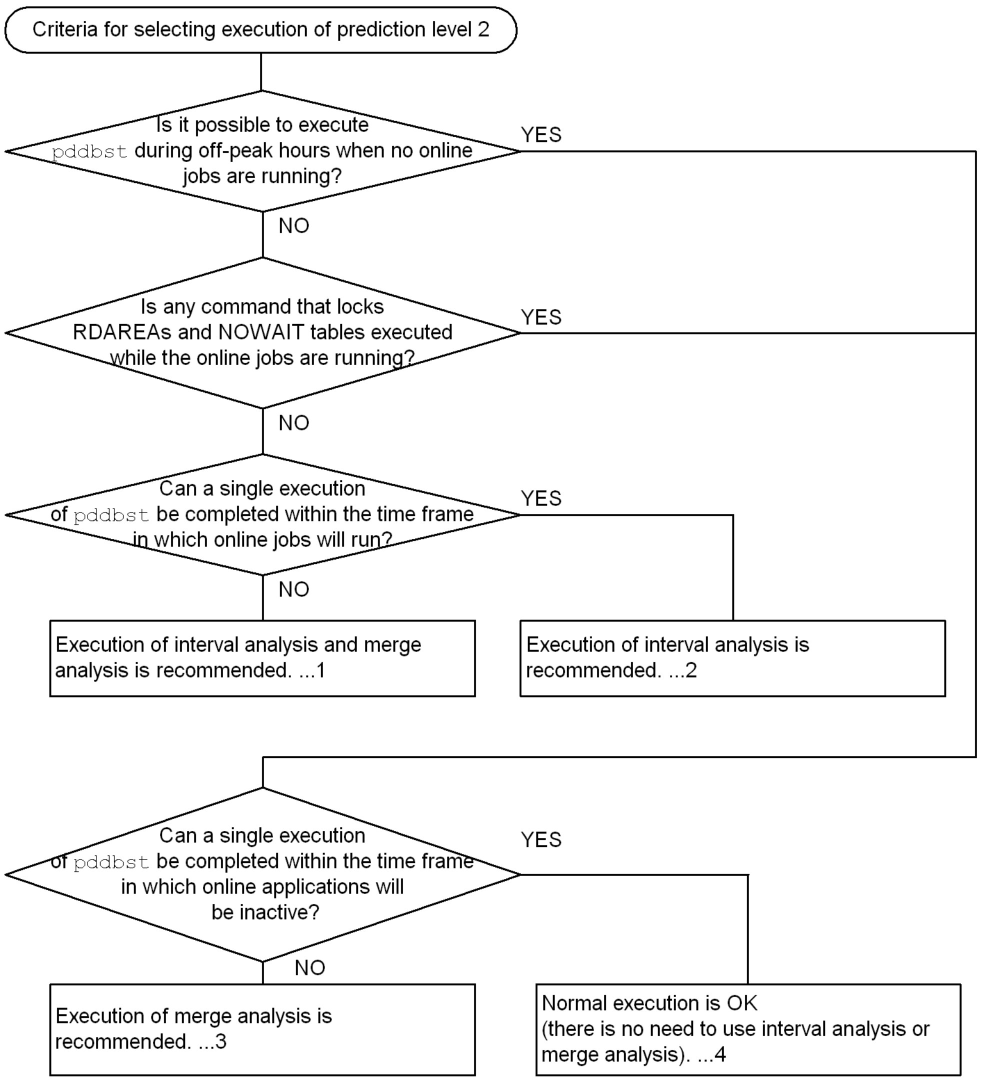
Figure 15-14: Overview of merge analysis


When the condition analysis result accumulation facility is executed 3 times (-n 3)
- First execution of merge analysis
- Second execution of merge analysis
- Third execution of merge analysis
Legend:
: Segment to be analyzed
: Segment that has already been analyzed
: Segment that has not been analyzed yet

15.6.3 Selection criteria

How the facility should be executed depends on the characteristics of the application. Figure 15-15 shows the execution method and selection criteria.

Figure 15-15: Execution method and selection criteria


Criteria for selecting execution of prediction level 2
Is it possible to execute pddbst during off-peak hours when no online jobs are running?
YES
NO
Is any command that locks RDAREAs and NOWAIT tables executed while the online jobs are running?
YES
NO
Can a single execution of pddbst be completed within the time frame in which online jobs will run?
YES
NO
Execution of interval analysis and merge analysis is recommended. ...1
Execution of interval analysis is recommended. ...2
Can a single execution of pddbst be completed within the time frame in which online applications will be inactive?
YES
NO
Execution of merge analysis is recommended. ...3
Normal execution is OK (there is no need to use interval analysis or merge analysis). ...4

Explanation

1. The following shows an example of command execution when interval analysis and merge analysis are used:

```
pddbst -r ALL -e 2 -w 100,100 -n 5
```

Explanation:

To minimize the effects on online jobs, the facility analyzes 100 segments of data and then pauses for 100 milliseconds, then repeats this process. This example executes the facility for 5 days to reduce the execution time per day. This command is executed once a day for 5 days.

2. The following shows an example of command execution when interval analysis is used:

```
pddbst -r ALL -e 2 -w 100,100
```

Explanation:

To minimize the effects on online jobs, the facility analyzes 100 segments of data and then pauses for 100 milliseconds, then repeats this process.

3. The following shows an example of command execution when merge analysis is used:

```
pddbst -r ALL -e 2 -n 5
```

Explanation:

This example reduces the execution time per day by executing `pddbst` over 5 days. The command in this example is executed once a day for 5 days.

4. The following shows an example of command execution during normal operation (neither interval analysis nor merge analysis is used):

```
pddbst -r ALL -e 2
```

Explanation:

This is the normal execution method where a single execution of `pddbst` completes the processing.

15.7 Rules and notes

(1) Rules

1. `pddbst` can be executed only while HiRDB is HiRDB is running.
2. Execute `pddbst` at the server machine that contains the single server or where the system manager is located.
3. The maximum number of concurrent executions of `pddbst` depends on the `pd_utl_exec_mode` operand value.

   When `pd_utl_exec_mode=0` is specified:

   The maximum number of concurrent executions of `pddbst` is 16.

   When `pd_utl_exec_mode=1` is specified:

   The maximum number of concurrent executions of `pddbst` is equal to the `pd_max_users` operand value.
4. An RDAREA to be analyzed (data dictionary RDAREA, data dictionary LOB RDAREA, registry RDAREA, or registry LOB RDAREA) must be in one of the following statuses:
   - Open status without shutdown
   - Command shutdown and open status
   - Referencing-possible shutdown and open status

   For details about user RDAREAs and user LOB RDAREAs, see Appendix *C. RDAREA Status During Command Execution*.
5. You must set the `LANG` environment variable in order to execute `pddbst`. You must set the `PDLANG` environment variable in order to use in the `pddbst` execution environment a character encoding that is not supported by the OS. For details about `LANG` and `PDLANG`, see the manual *HiRDB Version 8 UAP Development Guide*.

(2) Notes

1. The following are the `pddbst` utility's return codes:

   `0`: Normal termination

   `4`: Warning termination (invalid specification or skipped analysis processing)

   `8`: Abnormal termination
2. When `pddbst` is executed, an RDAREA to be analyzed is locked (shared retrieval mode: `SR`).

3. If logical analysis (`-k logi` specified) or analysis by table (`-t` specified) is performed on a table whose reorganization by `pdrorg` terminated abnormally, `DATA_UNFINISH` is displayed for `Status`.
4. When you execute the facility for predicting reorganization time, make sure that the `pd_lck_pool_size` operand value specified in the single server definition or the dictionary server definition is no less than the default value. Specify at least the value that is obtained from the following formula:

   `pd_lck_pool_size` operand value = $\uparrow y/x \uparrow$ (kilobytes)

   x: 6 in the 32-bit mode; 4 in the 64-bit mode

   y: Number of locked resources used by `pddbst`

   y=`MAX(`a`,`b`,`c`)`

   a: Number of defined RDAREAs $\times$ 30

   b: Number of defined tables + number of defined abstract data types + total number of columns of an abstract data type* + 61

   c: Number of defined indexes + 124

   * An abstract data type is counted once for each table or column in which it is used.

(3) Using files with a BOM

If you selected `utf-8` as the character encoding in the `pdsetup` command, you may be able to use a file with a BOM as the input file for `pddbst`. Table 15-9 shows whether or not files with a BOM can be used with `pddbst`. Note that even when a file with a BOM is used as the input file, the BOM is skipped. No BOM is included in the file that is output by `pddbst`.

Table 15-9: Whether or not files with a BOM can be used in pddbst (applicable to UTF-8)

Option	Input file	Use of file with a BOM
`-c`	Standard value definition file	Y
`-v`	Control statements file	Y

Legend:

Y: Can be used

15.8 Examples

The section shows an example of the use of `pddbst`.

(1) Database condition analysis facility

Example 1

This example performs condition analysis by table. The table to be analyzed is `STOCK`.

The example assumes that the `STOCK` table has been defined as shown in the example for `pddef`.

Command execution

```
pddbst -t STOCK
```

(2) Condition analysis result accumulation facility

Example

This example executes the condition analysis result accumulation facility on an RDAREA named `RD001` under the following conditions:

- Prediction level 2
- Merge analysis (3 times)

Command execution

```
 pddbst -r RD001 -e 2 -n 3
```

Note

This command is executed three times.

(3) Facility for predicting reorganization time

Example

This example executes the facility for predicting reorganization time in prediction level 2 based on the analysis information accumulated in (2).

Command execution

```
pddbst -k pred -e 2
```

Chapter

16. Optimizing Information Collection Utility (pdgetcst)

This chapter explains the optimizing information collection utility (`pdgetcst`), which collects information for optimizing based on cost and stores the information in the dictionary table.

This chapter contains the following sections:

16.1 Overview

Executor: User with the SELECT privilege for the target table

The optimizing information collection utility collects information required in order to optimize the system based on cost and registers the information in a data dictionary table. Registered optimization information can also be deleted from the data dictionary table. The optimizing information collection utility provides the following functions:

- Function for collecting optimization information by retrieval
- Function for registering optimization information from an optimization information parameter file

Conditions for executing the optimizing information collection utility

You should execute the optimizing information collection utility when all the following conditions are satisfied:

- The cost-based optimization mode is 2.[1]
- No application that involves updating operations is running.[2]
- There are many complex searches, such as join searches.
- There exists a test environment with the same data as in the real environment.[3]

1

If the cost-based optimization mode is not 2, you can still execute the utility if the optimizing information collection level is `lvl2`.

2

If an application that involves updating operations is running, but the optimizing information collection level is `lvl1` and the number of rows in the table has not changed, you can use the utility.

3

If there is no test environment with the same data as in the real environment, but you register the optimization information using a parameter file that contains optimized information, you can use the utility.

16.1.1 Collecting optimization information by retrieval

Determination of the optimum access path based on the database's current status is called cost-based optimization.

This enables HiRDB to use the most efficient access path and enables users to create UAPs without having to take SQL optimization into account.

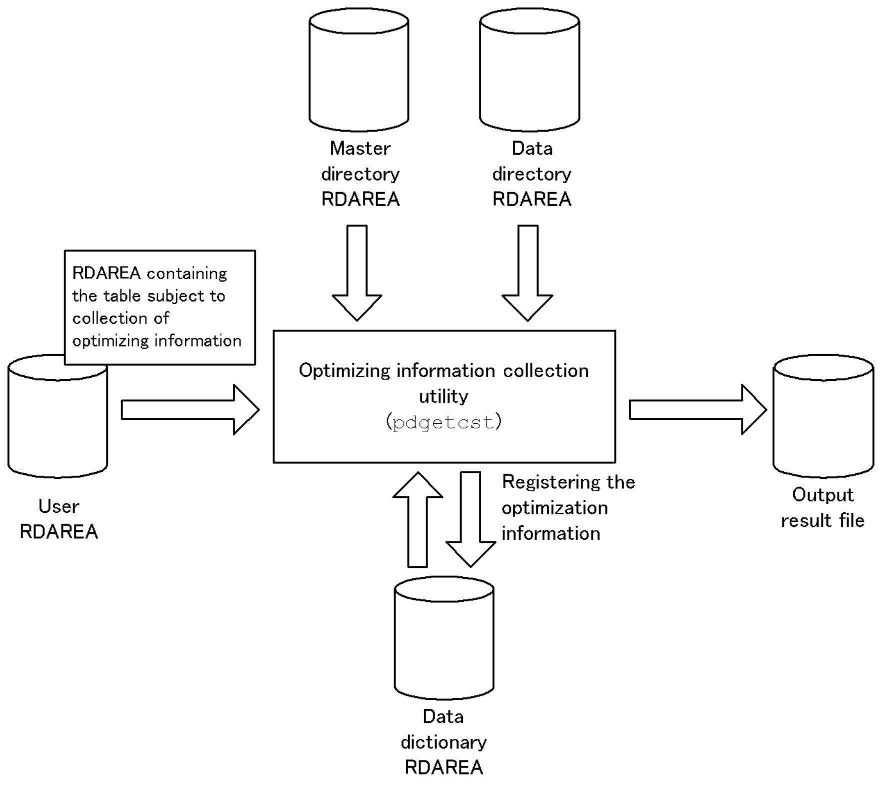
To optimize the system based on cost and to improve the precision of optimization, you must execute the optimizing information collection utility as needed before you compile a UAP or execute dynamic SQL statements.

You should collect optimization information by search after performing the following operations:

- After adding, updating, or deleting a large amount of data
- After executing the database load utility (`pdload`) or database reorganization utility (`pdrorg`)
- After adding indexes
- After executing the rebalancing utility

Figure 16-1 provides an overview of the optimizing information collection utility (`pdgetcst`).

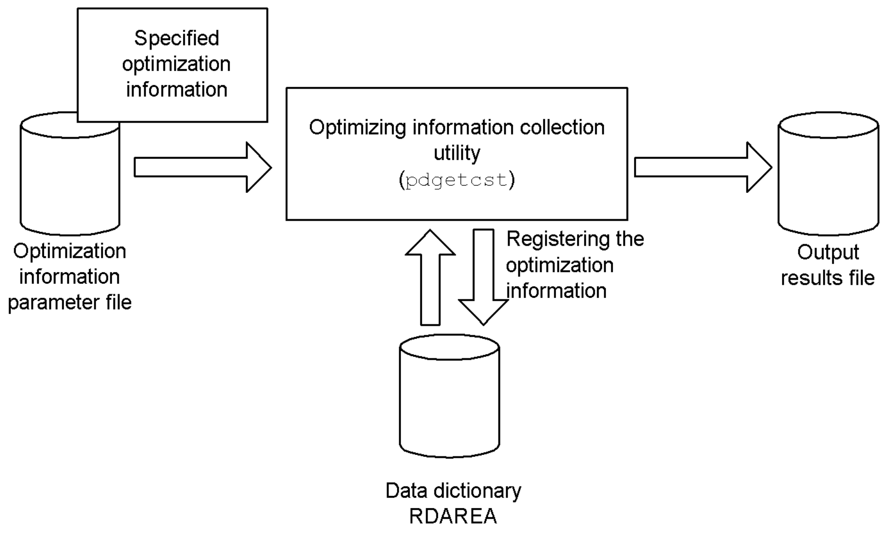
Figure 16-1: Overview of the optimizing information collection utility (pdgetcst)


Master directory RDAREA
Data directory RDAREA
RDAREA containing the table subject to collection of optimizing information
Optimizing information collection utility (pdgetcst)
User RDAREA
Registering the optimization information
Output result file
Data dictionary RDAREA

16.1.2 Registering optimization information from an optimization information parameter file

This function enables optimization information set in a parameter file to be registered in a dictionary table without having to collect it from the current database status. Figure 16-2 provides an overview of the function for registering optimization information from an optimization information parameter file of the optimizing information collection utility (`pdgetcst`).

Figure 16-2: Overview of the optimizing information collection utility (pdgetcst): registering optimization information from the optimization information parameter file


Specified optimization information
Optimization information parameter file
Optimizing information collection utility (pdgetcst)
Output results file
Registering the optimization information
Data dictionary RDAREA

16.1.3 Maximum number of concurrently executable utilities

The maximum number of copies of the optimizing information collection utility (`pdgetcst`) that can be executed concurrently depends on a value specified in the `pd_utl_exec_mode` operand of the system common definition:

`pd_utl_exec_mode=0`

A maximum of 16 copies can be executed concurrently.

`pd_utl_exec_mode=1`

The maximum number of copies of the utility that can be executed concurrently is equal to one half of the value specified in the `pd_max_users` operand.

16.2 Command format

16.2.1 Format

Collection of optimization information by retrieval or deletion of optimization information

```
pdgetcst [-u authorization-identifier] [-a authorization-identifier]
         [-p password] -t {table-identifier|ALL}
         [-d][-c {lvl1|lvl2}]
         [-l output-result-file]
         [-q generation-number]
```

Registration of optimization information from optimization information parameter file

```
pdgetcst [-u authorization-identifier] [-a authorization-identifier]
         [-p password] -t table-identifier
          -s optimization-information-parameter-filename
         [-l output-result-filename]
         [-v space-conversion-level]
```

Table 16-1 shows whether or not the options are specifiable depending on the specification of the `-c`, `-d`, or `-s` option.

Table 16-1: Whether or not options are specifiable depending on the specification of -c, -d, or -s option

Option		-c option		-d option	-s option
		lvl1	lvl2		
`-u` *authorization-identifier*		O	O	O	O
`-a` *authorization-identifier*		O	O	O	O
`-p` *password*		O	O	O	O
`-t`	*table-identifier*	R	R	R	R
	`ALL`		N		N
`-d`		N	N	—	N

Option		-c option		-d option	-s option
		lvl1	lvl2		
`-c`	`lvl1`	—	N	N	N
	`lvl2`	N	—	N	N
`-s` *optimization-information-parameter-filename*		N	N	N	—
`-l` *output-result-filename*		O	O	O	O
`-q` *generation-number*		O	O	O	O
`-v` *space-conversion-level*		N	N	N	O

R: Required

O: Optional

N: Cannot be specified

—: Not applicable

16.2.2 Options

- `-u` *authorization-identifier*

  Specifies the authorization identifier used for connecting to HiRDB. If omitted, the value set in the `PDUSER` environment variable is assumed. If the `PDUSER` value is not set, the user name corresponding to the user ID in the OS that is being used to execute this utility is assumed.

  If a user ID is enclosed in double quotation marks, it is handled as case sensitive; otherwise, it is handled as all uppercase letters. If you use the Bourne shell (sh), C shell (csh), or Korn shell (ksh), it is also necessary to enclose the user ID in single quotation marks.

- `-a` *authorization-identifier*

  Specifies the authorization identifier of the owner of the table that is to be subject to collection of optimization information.

  If omitted, the authorization identifier assigned by the `-u` option is assumed.

  If an authorization identifier is enclosed in double quotation marks, it is handled as case sensitive; otherwise, it is handled as all uppercase letters. If the Bourne shell (sh), C shell (csh), or Korn shell (ksh) is used, it is also necessary to enclose the authorization identifier in single quotation marks.

- `-p` *password*

  Specifies the password for the user ID specified in the `-u` option.

If the `-u` option is specified but the `-p` option is omitted, the user is requested to enter a password, and the entered value is used; if entry of a password is not required, the **NULL SEND** button should be pressed.

If the `-u` and the `-p` options are both omitted, the value set in the `PDUSER` environment variable is assumed; if the `PDUSER` value is not set, the user is requested to enter a password, and the entered value is used; if entry of a password is not required, the **NULL SEND** button should be pressed.

This utility must not be executed in an environment in which a required password cannot be entered, such as background execution with a shell using `&`. Password entry is required in the following cases:

- `-u` option specified and `-p` option omitted
- `-u` and `-p` options both omitted and no password set in the `PDUSER` environment variable

If a password is enclosed in double quotation marks, it is handled as case sensitive; otherwise, it is handled as all uppercase letters. If the Bourne shell (sh), C shell (csh), or Korn shell (ksh) is used, it is also necessary to enclose the password in single quotation marks.

■ `-t` {*table-identifier*|`ALL`}

table-identifier

Specifies the identifier of the table that is to be subject to collection of optimization information.

You cannot specify a data dictionary table, view table, or foreign table. However, when registering optimizing information using a parameter file that contains optimized information, you can specify a foreign table.

If a table identifier is enclosed in double quotation marks (`"`), the command treats it as being case sensitive. If it is not enclosed in double quotation marks (`"`), the command treats it as in all uppercase letters. If a table identifier contains a space, enclose the identifier in double quotation marks (`"`). If you are using `sh` (Bourne shell), `csh` (C shell), and `ksh` (Korn shell), you need to enclose the entire identifier in single quotation marks (`'`).

`ALL`

Specifies that all tables in the schema are to be subject to processing. Note that foreign tables cannot be subject to processing.

■ `-d`

Specifies that existing optimization information for the table specified with the `-t` option and for indexes defined for that table is to be deleted.

■ `-c {lvl1|`<u>`lvl2`</u>`}`

Specifies the optimizing information collection level.

For details about the optimizing information collection level, see Section *16.3.2 Optimizing information collection levels*.

`lvl1`

Collects the number of rows in the table and the statistical information cache size.

`lvl2`

Collects all optimizing information for the table.

- `-s` *optimization-information-parameter-filename* ~ <pathname>

  Specifies the absolute pathname of the optimization information parameter file containing the optimization information. When this option is specified, the utility registers the optimization information from the specified parameter file without collecting it from the current database status. When this option is specified, the `-d` option cannot be specified.

- `-l` *output-result-filename* ~ <pathname>

  Specifies the absolute pathname of the file to which the result of the optimizing information collection utility is to be output. When this option is omitted, the execution result is not output.

  If you specify `-t ALL` along with this option, and `pdgetcst` results in an error, you can identify the table resulting in the error.

- `-q` *generation-number* ~ <unsigned integer> ((0-10))

  When an RDAREA with replica RDAREAs is to be subject to collection of optimization information, specifies the generation of the target RDAREA.

  When this option is omitted, the command assumes the current RDAREA. The command also assumes the current RDAREA if the specified generation has no replica RDAREA.

  An error results if this option is specified when the inner replica facility is not used. Specifying this option also results in an error if there is no replica RDAREA of the index storage RDAREA or LOB column storage RDAREA that corresponds to the table storage RDAREA with the specified generation.

- `-v` *space-conversion-level*

  Specifies whether or not to convert spaces in the maximum and minimum column values.

  When this option is omitted, the utility executes space conversion according to the `pd_space_level` operand specification in the system common definitions.

For the space conversion level, specify `0`, `1`, or `3`:

`0`

Space conversion is not executed.

`1` or `3`

Space conversion is executed. `1` and `3` have the same effects.

If the table columns to be registered have the national character string type or mixed character string type, the utility converts spaces in the corresponding optimizing parameter file as follows:

- Column of national character string type

  The utility converts two consecutive single-byte spaces to one double-byte space, in units of two bytes from the top.

- Column of mixed character string type

  The utility converts each double-byte space to two single-byte spaces.

  When the character codes are `utf-8`, the system converts one double-byte space (3 bytes) to two single-byte spaces. For `MCHAR`, the system adds trailing single-byte spaces up to the definition length. For `MVARCHAR`, the data length remains shortened.

16.3 Optimizing information

16.3.1 Optimization information to be collected

Table 16-2 lists the optimization information that can be collected by the optimizing information collection utility or that can be registered using the optimization information parameter file.

Table 16-2: Optimization information collected by the optimizing information collection utility or registered using the optimization information parameter file

No.	Information collected	Classification	Description	SQL statement that improves precision of optimization	Processing that has adverse effects on integrity between registered optimization information and database status	Item name*
1	Number of rows	T	Total number of rows in the target table	SQL statement containing join	SQL statement that involves an updating operation and that changes the number of rows in the target table	`NROWS`
2	Number of data pages	T	Total number of row data storage pages in the target table	SQL statement that contains columns comprising the index in the search condition	SQL statement that involves an updating operation	`NPAGES`
3	Number of unique index key values	I	Null value (when the optimization information parameter file is used, the specified value is registered)			`NENTRY`

No.	Information collected	Classification	Description	SQL statement that improves precision of optimization	Processing that has adverse effects on integrity between registered optimization information and database status	Item name*
4	Number of index pages	I	Total number of index storage pages used			`NIPAGES`
5	Number of index levels	I	Maximum number of index levels in each single server or back-end server			`NLEVEL`
6	Sequentialness	I	Degree of sequentialness of the row data stored in order of index key values			`SEQ_RATIO`
7	Number of null values	C	Number of null values in the columns	SQL statement that contains a column comprising a single-column index or the first column comprising a multicolumn index		`NNULLS`
8	Maximum number of duplicate column values	C	Maximum number of duplicate values for a column			`N_MAX_DUP_KEY`
9	Minimum number of duplicate column values	C	Minimum number of duplicate values for a column			`N_MIN_DUP_KEY`
10	Column value distribution information	C	Column value distribution information			`MAX_VALUE`, `MIN_VALUE`

No.	Information collected	Classification	Description	SQL statement that improves precision of optimization	Processing that has adverse effects on integrity between registered optimization information and database status	Item name*
11	Number of unique column values	C	Number of unique column values			`NUNIQUE`
12	Statistical information cache size	T	Work area used by the HiRDB system	—	—	—

T: Table information.

I: Index information.

C: Column information:

- When collecting optimization information by retrieval

  This is the information on columns comprising an index; it is collected when an index is defined. In the case of a multicolumn index, only the information on the first column is collected.

- When registering optimization information from the optimization information parameter file

  This is the information on the columns specified in the optimization information parameter file.

*This item name is specified in the optimization information parameter file. For details about the item names that are specified in the optimization information parameter file, see *16.4.1 Information to be specified in the optimization information parameter file*.

Table 16-3 shows the information that is collected or registered as key value distribution information for index component columns.

Table 16-3: Information collected or registered as key value distribution information for index component columns

No.	Information collected	Description
1	Number of null values	All rows in the target table
2	Number of table rows	Total number of pages storing the row data of the target table
3	Column data length	Data length of the column
4	Column data type	Data type of the column
5	Number of sections	• Number of sections when the value of (number of table rows - number of rows whose key column data is null) is divided by a maximum of 100 • If the number of rows is less than 100, the number of rows is set. • Information in Nos. 8-10 is collected as many times as there are sections.
6	Maximum column value	Maximum value of the column
7	Minimum column value	Minimum value of the column
8	Degree of cumulation between sections	Total number of elements from section 1 through the corresponding section
9	Number of unique values in section	Number of unique column values in the corresponding section
10	Maximum value in section	Maximum column value in the corresponding section

16.3.2 Optimizing information collection levels

When collecting optimizing information by retrieval, you can specify the level of information to be collected. This is called optimizing information collection level.

Table 16-4 shows the advantages and disadvantages of each optimizing information collection level.

Table 16-4: Advantages and disadvantages of each optimizing information collection level

Optimizing information collection level	Advantage	Disadvantage
`lvl1`	• Optimization precision is comparatively high because this level lets you incorporate the number of rows in the table in the optimization and the data characteristics in the join order. • The execution time of `pdgetcst` is comparatively short.	• Data updating has effects on the optimization. • If the expected access path is not obtained, you must take into account the cost information on the number of rows in the table, thereby making it difficult to predict the access path.
`lvl2`	Optimizing precision is high because this level lets you incorporate the number of rows in the table, maximum and minimum values, and distribution information, in the optimization and data characteristics in the selection of join order and indexes.	• Data updating has effects on the optimization. • The `pdgetcst` execution time is long. • If the expected access path is not obtained, you must take into account all the cost information that has been acquired, thereby making it difficult to predict the access path. • If a multi-column index is defined, the optimizing precision is low because the utility obtains information only on the first index component column.

Check the characteristics of the table to be accessed and determine the appropriate optimizing information collection level. In some cases, it may be better to not obtain the optimizing information (to not execute `pdgetcst`).

If you are not collecting the optimizing information, there is no need to pay attention to the changes made to data by update-SQL. In this case, however, the optimizing precision is not high because the data characteristics are not incorporated in the optimization.

Table 16-5 shows the recommended optimizing information collection level.

Table 16-5: Recommended optimizing information collection level

No.	Characteristics of table to be accessed	Recommended optimizing information collection level
1	Both of the following conditions are satisfied: • No application that involves an updating operation is executed • The SQL statement to be executed contains columns comprising the index in the search condition.	`lvl2`

No.	Characteristics of table to be accessed	Recommended optimizing information collection level
2	All of the following conditions are satisfied: • Conditions in No. 1 do not apply. • Cost-based optimization mode is `2`. • An application that involves an updating operation and that changes the number of storage rows is not executed. • One of the following conditions is true: ● Hash join is performed. ● Hash execution of subquery is executed. ● Join search involving 3 or more tables is performed.	`lvl1`
3	Other	Do not collect optimizing information (do not execute `pdgetcst`).

16.4 Creating the optimization information parameter file

Before optimization information can be registered, the optimization information parameter file containing appropriate information must be created. Figure 16-3 shows the format of the optimization information parameter file.

Figure 16-3: Format of the optimization information parameter file

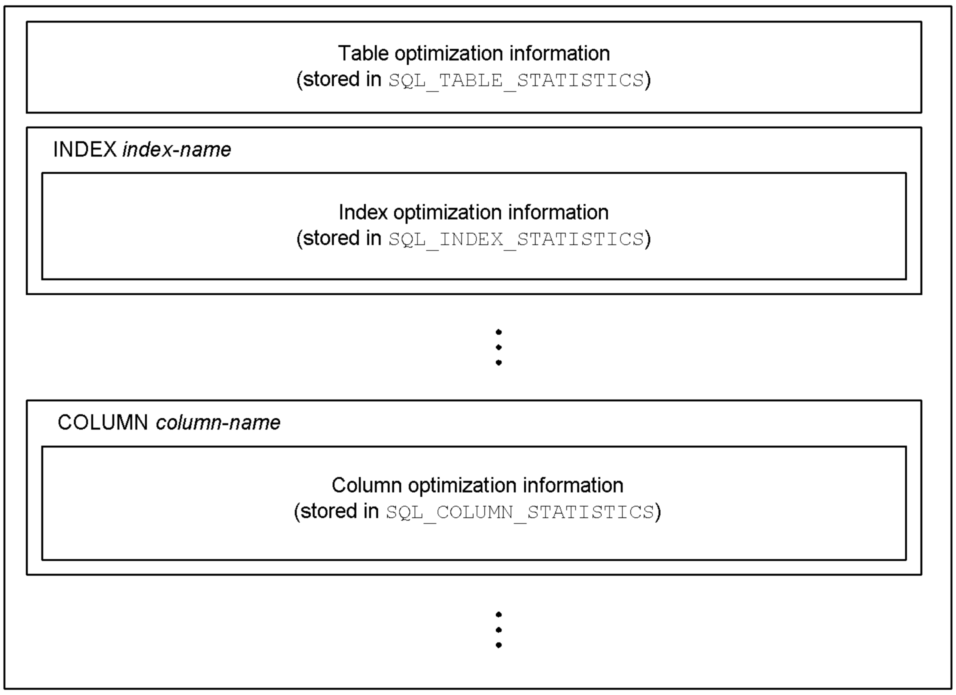


Table optimization information
(stored in SQL_TABLE_STATISTICS)
INDEX *index-name*
Index optimization information
(stored in SQL_INDEX_STATISTICS)
COLUMN *column-name*
Column optimization information
(stored in SQL_COLUMN_STATISTICS)

16.4.1 Information to be specified in the optimization information parameter file

Table 16-6 lists the items to be specified in the optimization information parameter file, and Table 16-7 provides a description of each item.

Table 16-6: Items specified in the optimization information parameter file

Classification	Item	Description
Table optimization information	`NROWS`	Number of rows in table
	`NPAGES`	Number of data pages in table

Classification	Item	Description
Index optimization information	`INDEX`	Index name
	`NIPAGES`	Number of index pages
	`NLEVEL`	Number of index levels
	`SEQ_RATIO`	Degree of index sequentiality
	`NENTRY`	Number of index key values
Column optimization information	`COLUMN`	Column name
	`NUNIQUE`	Number of unique column values
	`NNULLS`	Number of null values in column
	`N_MAX_DUP_KEY`	Maximum number of duplicate column values
	`N_MIN_DUP_KEY`	Minimum number of duplicate column values
	`MAX_VALUE`	Maximum column value
	`MIN_VALUE`	Minimum column value

Table 16-7: Descriptions of items in the optimization information parameter file

Item	Description
`NROWS`	Specifies the number of rows in the table. The following SQL statement can be used to obtain the number of rows in a table: `SELECT COUNT(*) FROM` *table-name*
`NPAGES`	Specifies the number of data pages used in the table. Execute the database condition analysis utility or see the *HiRDB Version 8 Installation and Design Guide*.
`INDEX`	Specifies the name of the index. The specified index must be defined in the table specified with the `-t` option.
`NIPAGES`	Specifies the number of index pages used for the index. Execute the database condition analysis utility or see the *HiRDB Version 8 Installation and Design Guide*.
`NLEVEL`	Specifies the number of levels for the index. See the *HiRDB Version 8 Installation and Design Guide*. In the case of a cluster key, this value can be obtained by the database condition analysis utility.

Item	Description
`SEQ_RATIO`	Specifies the degree of sequentiality of the index. Permitted value range is 0-100. If data is stored in the order of index key values, specify a value of 100; if data is stored in random order, specify a value of 0. As this value increases, the index is used more frequently.
`NENTRY`	Specifies the number of key values in the index: • For a unique index, specifies the number of rows in the table • For a non-unique index, specifies the number of key values in the index, excluding duplicate values The following SQL statements can be used to obtain the number of key values in the table: Single-column index: `SELECT COUNT(DISTINCT` *index-column-name*[1]`) FROM` *table-name* Multicolumn index: `WITH W1(WC1,WC2,...) AS (SELECT DISTINCT` *index-column-name-1*[1],*index-column-name-2*[1]`, ... FROM` *table-name*`) SELECT COUNT(*) FROM W1`
`COLUMN`	Specifies the name of a column. The specified column must be contained in the table specified with the `-t` option (However, LOB columns and columns of abstract data type cannot be specified.).
`NUNIQUE`	Specifies the number of unique values in the column. The following SQL statement can be used to obtain the number of unique values in a column: `SELECT COUNT(DISTINCT` *column-name*[2]`) FROM` *table-name*

`NNULLS`	Specifies the number of null values in the column.
	The following SQL statement can be used to obtain the number of null values in a column:
	• When the definition length of the column[3] is 32,000 bytes or less
	`SELECT COUNT(*) FROM` *table-name* `WHERE` *column-name* `IS NULL`
	• When the definition length of the column[3] is more than 32,000 bytes
	`WITH W1(WSUBCLM) AS (SELECT SUBSTR(`*column-name*`,1,255`[4]`)) FROM` *table-name*`)`
	`SELECT COUNT(*) FROM W1 WHERE WSUBCLM IS NULL`

Item	Description

N_MAX_DUP_KEY	Specifies the maximum number of duplicate column values in the table. The following SQL statement can be used to obtain the maximum number of duplicate column values:
	• When the definition length of the column^3 specified for the column name is 255 bytes or less
	WITH W1(WCOUNT) AS (SELECT COUNT(*column-name*) FROM *table-name*
	WHERE *column-name* IS NOT NULL GROUP BY *column-name*)
	SELECT MAX(WCOUNT) FROM W1
	• When the definition length of the column^3 specified for the column name is in the range 256-32,000 bytes
	SELECT COUNT(SUBSTR(*column-name*,1,255^4)) FROM *table-name*
	WHERE *column-name* IS NOT NULL GROUP BY SUBSTR(*column-name*,1,255^4)
	ORDER BY 1 DESC LIMIT 1
	• When the definition length of the column^3 specified for the column name is more than 32,000 bytes
	WITH W1(WSUBCLM) AS (SELECT SUBSTR(*column-name*,1,255^4)) FROM *table-name*)
	SELECT COUNT(WSUBCLM) FROM W1 WHERE WSUBCLM IS NOT NULL
	GROUP BY WSUBCLM ORDER BY 1 DESC LIMIT 1

N_MIN_DUP_KEY	Specifies the minimum number of duplicate column values in the table. The following SQL statement can be used to obtain the minimum number of duplicate column values:
	• When the definition length of the column^3 specified for the column name is 255 bytes or less
	WITH W1(WCOUNT) AS (SELECT COUNT(*column-name*) FROM *table-name*
	WHERE *column-name* IS NOT NULL GROUP BY *column-name*)
	SELECT MIN(WCOUNT) FROM W1
	• When the definition length of the column^3 specified for the column name is in the range 256-32,000 bytes
	SELECT COUNT(SUBSTR(*column-name*,1,255^4)) FROM *table-name*
	WHERE *column-name* IS NOT NULL GROUP BY SUBSTR(*column-name*,1,255^4)
	ORDER BY 1 ASC LIMIT 1
	• When the definition length of the column^3 specified for the column name is more than 32,000 bytes
	WITH W1(WSUBCLM) AS (SELECT SUBSTR(*column-name*,1,255^4)) FROM *table-name*)
	SELECT COUNT(WSUBCLM) FROM W1 WHERE WSUBCLM IS NOT NULL
	GROUP BY WSUBCLM ORDER BY 1 ASC LIMIT 1

Item	Description
`MAX_VALUE`[5]	Specifies the maximum column value. The following SQL statement can be used to obtain the maximum column value: `SELECT MAX(`*column-name*[6]`) FROM` *table-name*
`MIN_VALUE`[5]	Specifies the minimum column value. The following SQL statement can be used to obtain the minimum column value: `SELECT MIN(`*column-name*[6]`) FROM` *table-name*

[1] If the definition length of the column specified for the index column name is 256 bytes or more, replace *index-column-name* with `SUBSTR (`*index-column-name*`,1,255`[4]`)`.

[2] If the definition length of the column specified for the column name is 256 bytes or more, replace *column-name* with `SUBSTR (`*column-name*`,1,255`[4]`)`.

[3] Definition lengths are in bytes. For the national character string data type, the value is *definition length* × *2*.

[4] If the data type of the column is character string data or `BINARY`, the value is 255. If it is national character string or mixed character string, the value is 127.

[5] `MAX_VALUE` is stored in byte 32 of the `RANGE_VALUES` column in the `SQL_COLUMN_STATISTICS` data dictionary table. `MIN_VALUE` is stored in byte 48 of the `RANGE_VALUES` column in the `SQL_COLUMN_STATISTICS` data dictionary table.

[6] If the definition length of the column is 256 bytes or more, make the following replacement:

- If the data type of the column is `CHAR` or `BINARY`, replace *column-name* with `SUBSTR(`*column-name*`,1,255)`.
- If the data type of the column is `NCHAR` or `MCHAR`, replace *column-name* with `SUBSTR(`*column-name*`,1,127)`.
- If the data type of the column is `VARCHAR`, `NVARCHAR`, or `MVARCHAR`, replace *column-name* with the following specification:

```
  CASE LENGTH(column-name) WHEN  0 THEN ''
     WHEN  1 THEN SUBSTR(column-name,1,1)  WHEN  2 THEN
SUBSTR(column-name,1,2)
     WHEN  3 THEN SUBSTR(column-name,1,3)  WHEN  4 THEN
SUBSTR(column-name,1,4)
     WHEN  5 THEN SUBSTR(column-name,1,5)  WHEN  6 THEN
```

```
SUBSTR(column-name,1,6)
     WHEN  7 THEN SUBSTR(column-name,1,7)  WHEN  8 THEN
SUBSTR(column-name,1,8)
     WHEN  9 THEN SUBSTR(column-name,1,9)  WHEN 10 THEN
SUBSTR(column-name,1,10)
     WHEN 11 THEN SUBSTR(column-name,1,11)  WHEN 12 THEN
SUBSTR(column-name,1,12)
     WHEN 13 THEN SUBSTR(column-name,1,13) ELSE
SUBSTR(column-name,1,14) END
```

16.4.2 Specifications in the optimization information parameter file

Table 16-8 describes how to specify the items in the optimization information parameter file, and Table 16-9 describes how to specify `MAX_VALUE` and `MIN_VALUE`. Table 16-10 lists the default values for items in the optimization information parameter file.

Table 16-8: Specifications in the optimization information parameter file

Item	Specification	Permitted minimum value	Permitted maximum value	Units
`NROWS`	Specify a positive integer. An exponential value can also be specified.	0	Maximum `FLOAT` value supported by HiRDB	Rows
`NPAGES`		0		Pages
`INDEX`	Comply with the rules for index definition.			
`NIPAGES`	Specify a positive integer. An exponential value can also be specified.	1	Maximum `FLOAT` value supported by HiRDB	Pages
`NLEVEL`	Specify a positive integer.	1	Maximum `SMALLINT` value supported by HiRDB	Columns
`SEQ_RATIO`		0	100	%
`NENTRY`	Specify a positive integer. An exponential value can also be specified.	0	Maximum `FLOAT` value supported by HiRDB	Values
`COLUMN`	Comply with the rules for table definition.			
`NUNIQUE`	Specify a positive integer. An exponential value can also be specified.	0	Maximum `FLOAT` value supported by HiRDB	Values
`NNULLS`		0		Values
`N_MAX_DUP_KEY`		0		Values
`N_MIN_DUP_KEY`		0		Values

Item	Specification	Permitted minimum value	Permitted maximum value	Units
MAX_VALUE	—			
MIN_VALUE				

—: Not applicable.

Table 16-9: Specification of MAX_VALUE and MIN_VALUE

Data type of column specified for COLUMN	Specification	Maximum and minimum values
INT	Specify a numeric value.	HiRDB rules for each data type are observed. The permitted maximum number of digits complies with the HiRDB rules for numeric literals.
SMALLINT		
[LARGE]DEC[IMAL]		
FLOAT or DOUBLE PRECISION	Specify a numeric value. An exponential value can also be specified.	
SMALLFLT or REAL		
CHAR	Specify a character or a character string. If the character string contains a blank or its length is zero, enclose it in double quotation marks (").	One line of character string data including the item name can be specified in the optimization information parameter file. The utility registers the first 16 bytes as the optimization information (not including double quotation marks) and discards any remaining characters. If the data type is MCHAR or MVARCHAR and byte 16 or 17 contains a 2-byte code, at least 17 bytes of character string data must be specified.
VARCHAR		
NCHAR		
NVARCHAR		
MCHAR		
MVARCHAR		

Data type of column specified for COLUMN	Specification	Maximum and minimum values
`DATE`	Specify in the format *YYYY-MM-DD*.	HiRDB rules for each data type are observed.
`TIME`	Specify in the format *hh:mm:ss*.	
`INTERVAL YEAR TO DAY`	Specify in the format *(±)YYYYMMDD*.	
`INTERVAL HOUR TO SECOND`	Specify in the format *(±)hhmmss*.	
`TIMESTAMP`	"*YYYY*-*MM*-*DD*Δ*hh*:*mm*:*ss*[.*nnnnnn*]" Δ : Single-byte space For the fraction part [.*nnnnnn*], specify 0, 2, 4, or 6 according to the definition.[1] Enclose the specified value in double quotation marks.	
`BINARY`	Specify binary data expressed as hexadecimal characters.[2] For data with a length of 0, specify `X''`.	The utility registers the first 12 bytes as the optimization information and discards any remaining characters.

[1] The following table describes the storage method when the decimal places are specified for the second part of the `TIMESTAMP` type:

Defined length	Decimal places in the second part of input data				
	0	2	4	6	Other
0	Stored as is	Truncated at the defined length			Error (`KFPL31002-E`)
2	Zeros are padded up to the defined length	Stored as is	Truncated at the defined length		
4	Zeros are padded up to the defined length		Stored as is	Truncated at the defined length	
6	Zeros are padded up to the defined length			Stored as is	

[2] Hexadecimal representation is `x'****'` or `X'****'`. For `****`, specify a hexadecimal value in units of two characters (0-9, a-f, and A-F).

Table 16-10: Default values for items in the optimization information parameter file

Item		Value	Name of target dictionary table	Name of target column	Set value	Rows subject to setting in target dictionary table
NROWS*	Specified	Specified	SQL_TABLE_STATISTICS	N_ROW	Value specified for NROWS	Table specified with -t option
		Not specified			Null value	
	Not specified	—			Value existing before update	
NPAGES	Specified	Specified		N_PAGE	Value specified for NPAGES	
		Not specified			Null value	
	Not specified	—			Value existing before update	
NIPAGES	Specified	Specified	SQL_INDEX_STATISTICS	N_IXPG	Value specified for NIPAGES	Index specified with INDEX
		Not specified			Null value	
	Not specified	—			Value existing before update	
NLEVEL	Specified	Specified		N_LEVEL	Value specified for NLEVEL	
		Not specified			Null value	
	Not specified	—			Value existing before update	
SEQ_RATIO	Specified	Specified		SEQ_RATIO	Value specified for SEQ_RATIO	
		Not specified			Null value	
	Not specified	—			Value existing before update	

Item		Value	Name of target dictionary table	Name of target column	Set value	Rows subject to setting in target dictionary table
NENTRY	Specified	Specified		N_ENTRY	Value specified for N_ENTRY	
		Not specified			Null value	
	Not specified	—			Value existing before update	
NUNIQUE	Specified	Specified	SQL_COLUMN_STATISTICS	N_UNIQUE	Value specified for NUNIQUE	Column specified with COLUMN
		Not specified			Null value	
	Not specified	—			Value existing before update	
NNULLS*	Specified	Specified		N_NULL	Value specified for NNULLS	
		Not specified			Null value	
	Not specified	—			Value existing before update	
N_MAX_DUP_KEY	Specified	Specified		N_MAX_DUP_KEY	Value specified for N_MAX_DUP_KEY	
		Not specified			Null value	
	Not specified	—			Value existing before update	
N_MIN_DUP_KEY	Specified	Specified		N_MIN_DUP_KEY	Value specified for N_MIN_DUP_KEY	
		Not specified			Null value	
	Not specified	—			Value existing before update	

Item		Value	Name of target dictionary table	Name of target column	Set value	Rows subject to setting in target dictionary table
MAX_VALUE*	Specified	Specified		RANGE_VALUES	Value specified for MAX_VALUE	
		Not specified			Null value	
	Not specified	—			Null value	
MIN_VALUE*	Specified	Specified			Value specified for MIN_VALUE	
		Not specified			Null value	
	Not specified	—			Null value	

—: Not applicable.

* If either NROWS, NNULLS, MAX_VALUE, or MIN_VALUE is omitted, the null value is set in the RANGE_VALUES column.

16.4.3 Specification example of the optimization information parameter file

Following is a specification example of an optimization information parameter file:

```
# Table optimization information
NROWS            100      # Total number of rows in table
NPAGES           5        # Number of data pages in table
# Index optimization information
INDEX            IDX_I01  # Index name
NIPAGES          2        # Number of index pages
NLEVEL           2        # Number of index levels
SEQ_RATIO        80       # Degree of sequentiality of index
NENTRY           50       # Number of index key values
# Column optimization information
COLUMN           COL_01   # Column name
NUNIQUE          50       # Number of unique values in column
NNULLS           2        # Number of null values in column
N_MAX_DUP_KEY    5        # Maximum number of duplicate column
                            values
N_MIN_DUP_KEY    1        # Minimum number of duplicate column
                            values
MAX_VALUE        ZZZ      # Maximum column value
MIN_VALUE        ABC      # Minimum column value
```

16.4.4 Specification rules for the optimization information parameter file

- Begin the name of each item in byte 1and complete each item on one line.
- Separate an item name and its value with a blank (two-byte blank) or a tab.
- No more than one item can be specified on each line. One line consists of 80 bytes (not including linefeed code).
- Anything between the # character and the linefeed code is regarded as a comment.
- The same item cannot be specified more than once in the table, index, or column optimization information.
- If a specified value contains a blank or has a length of zero, it (the value) must be enclosed in double quotation marks ("). If a specified value contains a backslash (\), the data immediately following \ is used. To use \ as part of the data, two consecutive backslashes in succession must be specified.
- Items can be specified in any order within a range of index or column optimization information. A range of index or column optimization information begins with an index name or column name and ends when another index name or column name appears.
- If the same index name or column name is specified more than once, the last one specified is effective.

16.5 Rules and notes

(1) Rules

1. The optimizing information collection utility (`pdgetcst`) can be executed only when the HiRDB system is running.
2. The optimizing information collection utility (`pdgetcst`) should be executed on a single-server or a server machine with a system manager.
3. Columns of abstract data type cannot be specified in an optimizing information parameter file.
4. Only B-tree indexes can be specified in an optimizing information parameter file; plug-in indexes cannot be specified.
5. Optimizing information cannot be collected by searching tables for repetition columns, except for `-c lvl1`.
6. The optimizing information collected by `pdgetcst` becomes available for use during SQL optimization processing.
7. If the optimization information registered by `pdgetcst` does not match the actual database status, performance may become poor because the appropriate access path cannot be selected. If you have executed an SQL statement that changes the database status, execute `pdgetcst` to obtain the database status that matches the optimization information.
8. If an actual environment and a test environment both exist, make sure that the optimization information matches the actual number of table rows and distribution of data values. When the optimizing information collection level is `lvl1`, only the number of table rows must match.
9. To execute `pdgetcst`, the data dictionary RDAREA must be in shutdown release and open status.
10. Whether or not `pdgetcst` is executable on an RDAREA depends on the status of the index storage RDAREA for the table subject to `pdgetcst` processing. For details, see Appendix *C. RDAREA Status During Command Execution*.
11. For the output results file and parameter file that contains optimized information, you need to grant access privileges to the HiRDB administrator beforehand.
12. You must set the `LANG` environment variable in order to execute `pdgetcst`. You must set the `PDLANG` environment variable in order to use in the `pdgetcst` execution environment a character encoding that is not supported by the OS. For details about `LANG` and `PDLANG`, see the manual *HiRDB Version 8 UAP Development Guide*.

(2) Notes

1. The following are the `pdgetcst` utility's return codes:

   `0`: Normal termination

   `4`: Warning termination (warning occurred concerning data dictionary table manipulation)

   `8`: Abnormal termination
2. Do not change definitions of the target table or index when the optimizing information collection utility is being executed; if definitions are changed, the optimizing information collection utility may produce spurious results.
3. Inserting or deleting rows or updating data in the target table or index when the optimizing information collection utility is running can produce errors in the resulting optimizing information.
4. The optimizing information collection utility should be run when a table contains actual data. Correct cost information cannot be obtained if the optimizing information collection utility is executed on a table that does not contain any data, for example, immediately after the table was defined.
5. If the character encoding in the character strings coded in the optimizing information parameter file is different from the character encoding used in the HiRDB system, the results of the optimizing information collection utility cannot be guaranteed.
6. Do not use the collection (`-d` option not specified) and the deletion (`-d` option specified) of optimizing information simultaneously on the same table. If these options are specified simultaneously, the results of the operation of the optimizing information collection utility cannot be guaranteed.
7. If the optimizing information collection utility terminates with an error, the results of its operation cannot be guaranteed. If this happens, resolve the error by referencing any error messages that were output and rerun the optimizing information collection utility.
8. When `pdgetcst` is executed, the access procedure changes. If the actual data characteristics do not match the optimizing information, there may be adverse effects on the performance. Therefore, conduct a thorough test before executing `pdgetcst`.
9. If there are a test environment and a real environment, and optimizing information is to be collected in the real environment, make sure that both environments have the same number of data items and the same distribution of data values so that the same optimizing information is obtained as for the test environment.

(3) Using files with a BOM

If you selected `utf-8` as the character encoding in the `pdsetup` command, you may be able to use a file with a BOM as the optimization information parameter file for `pdgetcst`. Note that even when a file with a BOM is used as the optimization information parameter file, the BOM is skipped. No BOM is included in the file that is output by `pdgetcst`.

16.6 Output format

(1) Collecting optimization information by retrieval

(a) Optimizing information collection level lvl2

The following shows the execution results of the optimizing information collection utility (optimizing information collection by retrieval with `-c lvl2`):

```
**      pdgetcst                             1995-03-31 12:00
                                                     [1]
  **  TABLE INFORMATION  **
  AUTHORIZATION         : user-id [2]
  TABLE NAME            : TABLE01 [3]
  ROW COUNT             : 120     [4]
  DATA PAGES            : 5       [5]
  CACHE SIZE            : 0       [6]


  **  INDEX-COLUMN INFORMATION  **
  INDEX NAME            : Idx001  [7]
  INDEX PAGES           : 5       [8]
  INDEX LEVELS          : 3       [9]
  COLUMN NAME           : Clm1    [10]
  COLUMN TYPE           : INTEGER [11]
  SEQUENTIAL RATIO      : 100     [12]
  MAX VALUE             : 2500    [13]
  MIN VALUE             : 0       [14]
  UNIQUE KEYS           : 23      [15]
  NULL VALUE COUNT      : 20      [16]
  MAX DUPLICATE KEY     : 10      [17]
  MIN DUPLICATE KEY     : 1       [18]
  SECTION COUNT         : 39      [19]


    **  SECTION INFORMATION  **
    SECTION NO          :1  [20]
    TOTAL COUNT         :1  [21]
    UNIQUE KEYS         :1  [22]
    MAX VALUE           :0  [23]


    **  SECTION INFORMATION  **
    SECTION NO          :2
    TOTAL COUNT         :2
    UNIQUE KEYS         :1
    MAX VALUE           :10
    **  SECTION INFORMATION  **
    SECTION NO          :3
    TOTAL COUNT         :6
    UNIQUE KEYS         :1
    MAX VALUE           :10

             :


    **  SECTION INFORMATION  **
    SECTION NO          :39
    TOTAL COUNT         :100
    UNIQUE KEYS         :1
    MAX VALUE           :2500

**  pdgetcst ENDED    RETURN CODE = 0 [24]
```

Explanation

`**  SECTION INFORMATION  **` is displayed as many times as there are sections. `**  INDEX-COLUMN INFORMATION  **` (including `**  SECTION INFORMATION  **`) is displayed as many times as there are indexes.

1. Date and time the optimizing information collection utility was executed (`pdgetcst` command entry time) in the format *YYYY/MM/DD hh:mm*.

   YYYY: Year. *MM*: Month. *DD*: Date. *hh*: Hour. *mm*: Minute.
2. Authorization identifier of the table subject to collection of optimization information.
3. Identifier of the table subject to collection of optimization information.
4. Number of rows in the table.
5. Number of data pages used in the table.
6. Work area used by HiRDB.
7. Name of an index subject to collection of optimization information.
8. Number of index pages used for the index.
9. Number of index levels.
10. Name of the first component column of the index.
11. Data type of the first component column of the index.
12. Sequentialness.
13. Maximum column value in the first component column of the index.
14. Minimum column value in the first component column of the index.

15. Number of unique values in the first component column of the index.
16. Number of `NULL` values in the first component column of the index.
17. Maximum number of duplicate values in the first component column of the index.
18. Minimum number of duplicate values in the first component column of the index.
19. Number of sections for section distribution information.
20. Section number.
21. Cumulative number of data items for the section.
22. Number of unique values in the section.
23. Maximum value in the section.
24. Return code.

Note

When the `-d` option is specified, 8, 9, and 12-19 are not displayed. If the number of rows is zero, 13 and 14 are left blank, and 0 is displayed for the other items.

(b) Optimizing information collection level lvl1

The following shows the execution results of the optimizing information collection utility (optimizing information collection by retrieval with `-c lvl1`):

```
**      pdgetcst                             2000-03-31 12:00
                                                     [1]

  **  TABLE INFORMATION  **
  AUTHORIZATION         : user-id [2]
  TABLE NAME            : TABLE01 [3]
  ROW COUNT             : 120     [4]    [6]
  CACHE SIZE            : 0       [5]

**  pdgetcst ENDED    RETURN CODE = 0 [7]
```

Explanation

1. Date and time the optimizing information collection utility was executed (`pdgetcst` command entry time) in the format *YYYY/MM/DD hh:mm*.

   YYYY: Year. *MM*: Month. *DD*: Date. *hh*: Hour. *mm*: Minute.
2. Authorization identifier of the table for which optimization information is being collected.
3. Identifier of the table for which optimization information is being collected.
4. Number of rows in the table.

5. Work area used by the HiRDB.
6. Information about other tables (applicable when `-t ALL` is specified).
7. Return code.

(2) Registering optimization information using the optimization information parameter file

```
**     pdgetcst                               1997-10-30 15:06
                                                       [1]
  **  TABLE INFORMATION    **
  AUTHORIZATION       : MANUAL [2]
  TABLE NAME          : TBL01  [3]
  ROW COUNT           : 1      [4]
  DATA PAGES          : 1      [5]
  CACHE SIZE          : 2560   [6]


  **  INDEX INFORMATION  **
  INDEX NAME          : IDX_TBL01_C1 [7]
  INDEX PAGES         : 1            [8]
  INDEX LEVELS        : 1            [9]
  SEQUENTIAL RATIO    : 1            [10]
  INDEX KEY           : 1            [11]


  **  COLUMN INFORMATION  **
  COLUMN NAME         : C1   [12]
  COLUMN TYPE         : CHAR [13]
  UNIQUE KEYS         : 1    [14]
  NULL VALUE COUNT    : 1    [15]
  MAX DUPLICATE KEY   : 1    [16]
  MIN DUPLICATE KEY   : 1    [17]
  MAX VALUE           : "1"  [18]
  MIN VALUE           : "1"  [19]
  SECTION COUNT       : 100  [20]


**  pdgetcst ENDED    RETURN CODE = 0 [21]   **
```

Explanation

`**  COLUMN INFORMATION  **` is displayed as many times as there are columns. `**  INDEX INFORMATION  **` and `**  COLUMN INFORMATION  **` are displayed as many times as there are indexes.

1. Date and time optimizing information collection utility executed (`pdgetcst` command entry time), in the format *YYYY-MM-DD hh:mm*.

   YYYY: Year. *MM*: Month. *DD*: Date. *hh*: Hour. *mm*: Minute.
2. Authorization identifier of the table subject to registration of optimization information.

3. Identifier of the table subject to registration of optimization information.
4. Number of rows in the table.
5. Number of data pages used in the tale.
6. Work area used by the HiRDB system.
7. Name of an index subject to registration of optimization information.
8. Number of index pages used for the index.
9. Number of index levels.
10. Degree of sequentiality.
11. Number of key values for the index.
12. Name of column for which index is defined.
13. Data type of column for which index is defined.
14. Number of unique values in column for which index is defined.
15. Number of null values in column for which index is defined.
16. Maximum number of duplicate values in column for which index is defined.
17. Minimum number of duplicate values in column for which index is defined.
18. Maximum value in column for which index is defined.
19. Minimum value in column for which index is defined.
20. Number of sections for section distribution information.
21. Return code.

16.7 Examples

Example 1 shows an example of using the optimizing information collection utility.

(a) Example 1

Collect optimization information for the following table and output the result to the following output results file:

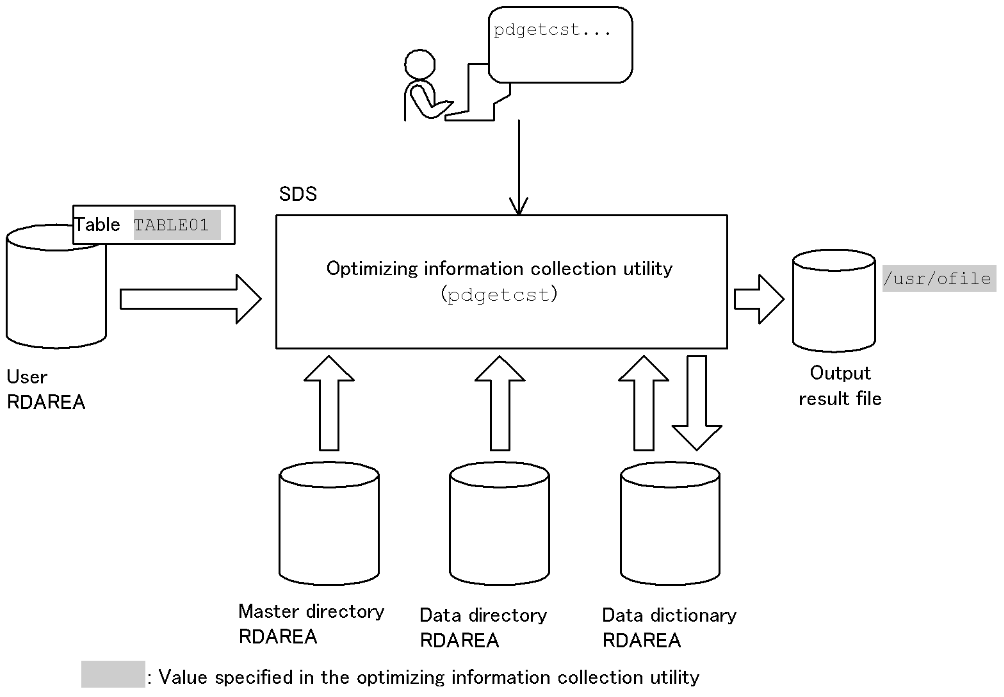
- Table identifier

  `TABLE01`

- Output results file

  `/usr/ofile`

Overview



Command execution

```
pdgetcst -a USER-ID   ........................1
         -t TABLE01   ........................2
         -l /usr/ofile   ....................3
```

Explanation:

1. Authorization identifier of the table subject to collection of optimization information: `USER-ID`
2. Identifier of the table subject to collection of optimization information: `TABLE01`
3. File for output of the execution result of the optimizing information collection utility: `/usr/ofile`

Chapter

17. Access Path Display Utility (pdvwopt)

This chapter explains the access path display utility (`pdvwopt`), which displays access path information that was determined by the SQL optimization processing.

This chapter contains the following sections:

17.1 Overview

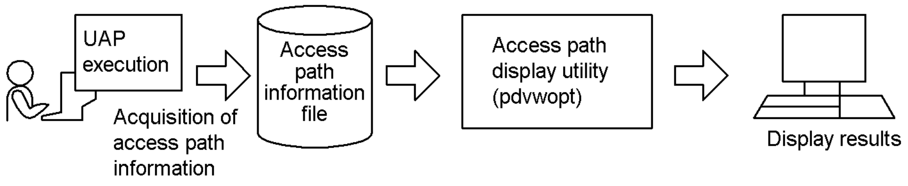
Executor: Any user

17.1.1 Function

HiRDB optimizes each SQL statement entered by the user in order to determine the optimum database access method (access path). The access path display utility (`pdvwopt`) displays the access path information obtained by this SQL optimization processing; the displayed information can be used for tuning purposes.

Figure 17-1 provides an overview of the access path display utility (pdvwopt).

Figure 17-1: Overview of the access path display utility (pdvwopt)


UAP execution
Acquisition of access path information
Access path information file
Access path display utility (pdvwopt)
Display results

17.2 Command format

17.2.1 Format

```
pdvwopt [-d] access-path-information-file
```

17.2.2 Option

- `-d`

  Specifies that join processing information, base table retrieval processing information, table IDs, and index IDs are to be displayed.

17.2.3 Argument

- *access-path-information-file* ~ <pathname>

  If you specify `1` or a greater value in the `PDVWOPTMODE` client environment definition variable, the utility stores the access path information in the access path information file.

  This argument specifies the access path information file that is to be subject to the execution of the access path display utility. For details about the `PDVWOPTMODE` client environment variable, see the *HiRDB Version 8 UAP Development Guide*.

  The access path information file is created under the SQL information directory (`$PDDIR/spool/pdsqldump`) at the unit that contains the single server or front-end server accessed by the UAP.

  Information is collected for each pair of `CONNECT` and `DISCONNECT` statements. HiRDB assigns the following name to each access path information file:

authorization-identifier_UAP-source-file-name_identification-number_server-type

authorization-identifier

Authorization identifier specified in the `CONNECT` statement.

UAP-source-file-name

Specifies the name of the UAP source file.

If the name is longer than 30 bytes, only the first 30 bytes are effective.

If the SQL was executed by the interactive SQL execution utility (`pdsql`), `pdsql`-*process-ID*, `pdsql.exel`-*process-ID*, or `pdsqlwl`-*process-ID* is used as the UAP source file name.

identification-number

Number used to identify the set of statements from `CONNECT` through `DISCONNECT`.

server-type

`s` for a HiRDB/Single Server; `p` for a HiRDB/Parallel Server.

Examples of access path information file names:

```
USER1_sample.ec_1-1-01_s
```

```
USER2_pdsql-00001_1-1-01_p
```

17.3 Execution procedure

To execute the access path display utility, use the following procedure:

Client

1. Set the client environment variables. For details about the client environment variables see the *HiRDB Version 8UAP Development Guide*.
2. Execute the UAP containing the SQL for which access path information is to be obtained. There is no need to complete the UAP's processing; the access path information file is created when the statements from `CONNECT` through `DISCONNECT` have executed.

Server

1. Execute the access path display utility at the server machine that contains the single server or front-end server accessed by the UAP.

Execution

```
$PDDIR/bin/pdvwopt $PDDIR/spool/pdsqldump/
USER1_pdsql-00001_1-1-02_p
```

17.4 Rules and notes

(1) Rules

1. The access path display utility can be executed at any time, whether or not HiRDB is active.
2. The access path display utility must be executed at the server machine containing the single server or front-end server that is accessed by the UAP subject to utility processing.

(2) Notes

(a) Notes about return code

For the `pdvwopt` utility, return code `0` indicates normal termination, and return code `8` indicates abnormal termination.

(b) Notes about collecting an access path information file

1. Specifying a value of `1` in the `PDVWOPTMODE` client environment definition variable increases the server's workload for creating an access path information file. Therefore, this value should be specified only when it is necessary to collect access path information. If output of access path information is specified with the UAP report facility, the server's workload may increase. For details about the UAP report facility, see the *HiRDB Version 8UAP Development Guide*.
2. If you specify a value of `1` in the `PDVWOPTMODE` client environment definition variable and if the SQL object being checked results in a cache hit, the utility uses the previously created SQL object because SQL optimization is not executed. Therefore, the utility does not obtain the access path information.
3. If you specify a value of `2` in the `PDVWOPTMODE` client environment definition variable, the utility obtains the access path information even when the SQL object being checked results in a cache hit. In this case, the utility creates SQL objects by executing SQL optimization regardless of the cache hit; therefore, the server's workload may be greater than when a value of `1` is specified in `PDVWOPTMODE`.
4. For an SQL statement specified in a stored procedure, executing the `CALL` statement does not obtain the access path information. In this case, one of the following actions must be taken in order to obtain access path information (a different access path might be obtained because the SQL object is re-created on the basis of current cost information):
   - Execute `ALTER PROCEDURE` to re-create the SQL object.
   - Delete the procedure with `DROP PROCEDURE`, then re-create the SQL object with `CREATE PROCEDURE`.

5. For a triggered SQL statement specified in a trigger, the utility does not obtain the access path information even if the trigger is executed. To obtain access path information for a triggered SQL statement, use the following method (note that the access path may be different from the previous path because the utility re-creates the SQL object on the basis of the current cost information):
   - Execute the `ALTER TRIGGER` statement to re-create the SQL object.
6. HiRDB does not delete access path information files. Any unneeded access path information files must be deleted by the user, taking care not to delete the SQL information directory (`$PDDIR/spool/pdsqldump`); if this directory is deleted, access path information files can no longer be created.
7. When you define in `CREATE TABLE` a referencing table for which `CASCADE` is specified as the referential constraint action, an internal trigger is created; therefore, access path information is acquired when such a referencing table is defined.
8. For the trigger created when a referencing table is defined, no access path information is acquired even when the trigger is executed. In this case, execute `ALTER ROUTINE` to re-create the SQL object and then acquire access path information. Note that the acquired access path may not match the existing access path because the SQL object is re-created on the basis of the current cost information.
9. If you execute the `INSERT` statement with `VALUES` specified, the utility acquires the access path information only when one of the following functions is executed:
   - Trigger
   - Referential constraint
   - Scalar subquery

(c) Notes about executing the access path display utility

The same version of the access path display utility and access path information file must be used; if their versions do not match, an error occurs.

(d) Notes on the output information in the output format

Internal system information in the execution results output by the access path display utility is enclosed in braces (`{ }`).

17.5 Output format

```
Version         :aa...a  bb...bBES
UAP Name        :cc...c
Authorization   :dd...d
-------------------------------------------------------
Section No      :ee...e
UAP Source      :ff...f
Optimize Time   :gggg-gg-gg  gg:gg:gg.gggggg
Optimize Mode   :hh...h
SQL Opt Level   :iiiiiiiiii(ii...i)  = jj...j(kk...k),...
Add Opt Level   :llllllllll(ll...l)  = mm...m(nn...n),...
Routine Name    :ss...s.ss...s
SQL             :oo...o
Work Table      :pp...p
Total Cost      :qq...q  (including rr...r FOREIGN SERVER SCAN)
Trigger Name    :tt...t
                    Procedure Name :uu...u
                    Action Time    :vv...v(ww...w)
                    Action Type    :xx...xx
----- QUERY EXPRESSION BODY ID : ... -----  ..................1
 :
----- QUERY ID : ... ----- ...................................2
 :
JOIN  ........................................................3
 :
SCAN  ........................................................4
 :
---------- FOREIGN SERVER(PREDICATION) : ... ----------  ......5
 :
```

Explanation

1. Information about set operation; for details, see *17.5.2 Set operation information*.
2. Information about query processing; for details see *17.5.3 Query processing information*.
3. Information about join processing; for details, see *17.5.4 Join processing information*.
4. Information about base table retrieval processing; for details, see *17.5.5 Base table retrieval processing information*.
5. Information about foreign server retrieval processing (prediction). This information is displayed for each retrieval request issued to foreign servers. For details about foreign server retrieval (prediction) information, see *17.5.1*

Foreign server retrieval processing (prediction) information.

Information following `Section No` is displayed as many times as there are SQL statements.

aa...a

Displays the HiRDB version. For a HiRDB update version, the version number is followed by a symbol.

bb...b

Displays the number of back-end servers for a HiRDB/Parallel Server. This information is not displayed for a HiRDB/Single Server.

cc...c

Displays the UAP name specified in the `PDCLTAPNAME` client environment variable.

dd...d

Displays the authorization identifier.

ee...e

Displays the section number (used to check the SQL statement correspondence).

ff...f

Displays the name of the UAP source file.

If the name is longer than 30 bytes, only the first 30 bytes are effective.

If the SQL was executed by the interactive SQL execution utility (`pdsql`), `pdsql`-*process-ID*, `pdsql.exel`-*process-ID*, or `pdsqlwl`-*process-ID* is used as the UAP source file name.

gggg-gg-gg gg:*gg*:*gg*.*gggggg*

Displays the date and time SQL optimization processing took place, in the format *year-month-date hour*:*minute*:*second.microsecond*.

hh...h

Displays the SQL optimization mode. For details about the SQL optimization mode, see the *HiRDB Version 8UAP Development Guide*.

- `COST_BASE_1`

  The utility applies the cost-based optimization mode 1.

- `COST_BASE_2`

  The utility applies the cost-based optimization mode 2.

iiiiiiiii (*ii...i*) = *jj...j* (*kk...k*) , . . .

Displays the value of the SQL optimization option during SQL compilation.

- *iiiiiiiii* (*ii...i*)

  Displays the value of the SQL option in hexadecimal and in decimal (the latter in parentheses).

- *jj...j* (*kk...k*) , . . .

  Displays the identifier of the function name specified in the SQL optimization option (unsigned integer). If more than one function name is specified, the utility separates each function by a comma.

lllllllll (*ll...l*) = *mm...m* (*nn...n*) , . . .

Displays the value of the extended SQL optimization option during SQL compilation.

- *lllllllll* (*ll...l*)

  Displays the value of the extended SQL optimization option in hexadecimal and in decimal (the latter in parentheses).

- *mm...m* (*nn...n*) , . . .

  Displays the identifier of the function name specified in the extended SQL optimization option (unsigned integer). If more than one function name is specified, the utility separates each function by a comma.

oo...o

Displays the compiled SQL statement.

pp...p

Displays the number of times a work table is created for the execution of the SQL statement. If no work table is to be created, the utility displays a value of 0.

If a sort operation does not take place, the utility creates only one work table. If a sort operation takes place, the utility creates two work tables.

Because the work table creation timing is not always the same, the execution of the SQL statement may require a smaller work table file than the estimated value. For details about how to estimate the work table file, see the *HiRDB Version 8Installation and Design Guide*.

If the query includes accesses to foreign tables, the utility displays the timing of creating work tables on the local HiRDB.

qq...q

Displays the total query cost obtained by HiRDB using cost-based optimization. This line is displayed only when the SQL optimization mode is cost-based optimization mode 2.

rr...r

If the query includes accesses to foreign tables, the utility displays the number of retrieval requests issued to foreign servers. If the query does not include access to foreign tables, this information is not displayed.

ss...s.ss...s

For an SQL statement in a routine, the utility displays the routine name in the format *authorization-identifier***.***routine-identifier*. If the SQL statement is not in a routine, this information is not displayed.

tt...t

Displays the name of the trigger to be executed by HiRDB. If there is no trigger to be executed, *tt...t* through *xx...x* are not displayed.

If there is more than one trigger to be executed, this information consists of multiple rows, in which case information is displayed in the order of execution.

uu...u

Displays the name of the trigger operation procedure.

vv...v

Displays the trigger operation timing:

`BEFORE`: Trigger to be executed before table manipulation

`AFTER`: Trigger to be executed after table manipulation

ww...w

For a trigger created when a referencing table is defined, this is the type of referential constraint action:

- `CASCADE`

  The referential constraint action is `CASCADE`.

xx...x

Displays the type of trigger operation units:

- `FOR EACH ROW`

  Trigger by row

- `FOR EACH STATEMENT`

Trigger by statement

The following is an example of the output results from execution of a trigger by row with `BEFORE` (`TRIGGER1`) as the trigger operation timing, a trigger by statement with `AFTER` (`TRIGGER2`) as the trigger operation timing, and a trigger created when the referential constraint action was `CASCADE` (`URA2004050521222324`):

```
Trigger Name   : TRIGGER1
                    Procedure Name : (TRIG2003030405060708)
                    Action Time    : BEFOR
                    Action Type    : FOR EACH ROW
                 (URA2004050521222324)
                    Procedure Name : (TRIG2004050521222324)
                    Action Time    : AFTER(CASCADE)
                    Action Type    : FOR EACH ROW
                 TRIGGER2
                    Procedure Name : (TRIG2003091011121314)
                    Action Time    : AFTER
                    Action Type    : FOR EACH STATEMENT
```

17.5.1 Foreign server retrieval processing (prediction) information

If a query includes accesses to foreign tables, access path information consists of information about the local HiRDB retrieval processing and information about the foreign server retrieval processing (prediction).

HiRDB first predicts the access path within the foreign server and obtains information about the foreign server retrieval processing (prediction). Next, based on this prediction, HiRDB determines information about the local HiRDB retrieval processing.

Information about the foreign server retrieval processing (prediction) is predicted by HiRDB in order to determine information about the local HiRDB retrieval processing; it is not the actual access path used for retrieval by the foreign DBMS. Information about the access path used by the foreign DBMS must be obtained at the foreign DBMS. If the foreign DBMS is a HiRDB, you can obtain the access path information file by executing an SQL statement with `1` or `2` specified in the `PDVWOPTMODE` operand of the foreign server definition.

```
--------- FOREIGN SERVER(PREDICTION) : aa...a ---------
 Derived Table   :bb...b
 Foreign Server :cc...c(dd...d) [ee...e ff...f]
 SQL             :gg...g
 Table Mapping   :hh...h(hh...h) [ii...i(ii...i)],...
 Var Mapping     :jj...j[kk...k],...
 Foreign Cost    :SERVER-ll...l,COMM-mm...m
 ----- QUERY EXPRESSION BODY ID : ... -----   ..........1
   :
 ----- QUERY ID : ... -----   ........................2
   :
 JOIN   ...............................................3
   :
 SCAN   ...............................................4
   :
```

Explanation

1. Set operation information; for details, see *17.5.2 Set operation information*.
2. Query processing information; for details, see *17.5.3 Query processing information*.
3. Join processing information; for details, see *17.5.4 Join processing information*.
4. Base table retrieval processing information; for details, see *17.5.5 Base table retrieval processing information*.

aa...a

Displays the access number of the query that includes an access to a foreign table. The access number is assigned sequentially to each foreign server retrieval processing (prediction) information item beginning at 1.

bb...b

Displays the internally created table name (`FOREIGNSQL` *table-number*) for the result of retrieval from the foreign server. The table number is a sequential 3-digit integer beginning at 1.

cc...c(*dd...d*) [*ee...e ff...f*]

Displays information about the foreign server and foreign DBMS.

cc...c

Displays the name of the foreign server.

dd...d

Displays the name of the back-end server assigned to access the foreign server.

ee...e

Displays the server type of the foreign DBMS.

ff...f

Displays the server version of the foreign DBMS that was specified when the foreign server was defined.

gg...g

Displays the SQL statement that is issued to the foreign server.

Some SQL statement components are displayed using the foreign DBMS format and some are displayed without using the foreign DBMS format.

Components that are displayed using the foreign DBMS format:

- Table identifier

  The table identifier is converted to the actual table identifier defined in the foreign DBMS. If the table identifier defined in the foreign server is different from the table identifier defined in the foreign DBMS, the utility displays their correspondence in `Table Mapping`.

- Variables

  Embedded variables, `?` parameters, SQL variables, SQL parameters, `USER`, `CURRENT_DATE`, `CURRENT_TIME`, and `CURRENT_TIMESTAMP` are converted to the foreign DBMS's variable format.

  If a specified condition includes a column of another foreign table, the utility may convert it to the variable depending on the join method.

  Correspondence of variables before and after conversion is displayed in `Var Mapping`.

  If the foreign DBMS is ORACLE, `:`*?-number* is displayed for variables (*?-number*: sequential number assigned to `:`*?-number*, which is in the SQL statement issued to the foreign server, from left to right beginning at 1).

Components that are displayed without using the foreign DBMS format:

When executing actual retrieval, the utility issues to the foreign server an SQL statement in the format usable by the foreign DBMS.

- Lock option

  The utility specifies the lock option as required. The lock option is displayed in the format usable by HiRDB.

- Update option

  The utility adds the update option (`FOR READ ONLY` and `FOR UPDATE` clauses) as required and displays it in the format usable by HiRDB.

hh...h`(`*hh...h*`)` `[`*ii...i*`(`*ii...i*`)]`, . . .

Displays the correspondence of the table identifier *hh...h* and correlation name (*hh...h*) contained in the SQL statement issued to the foreign server to the table identifier *ii...i* and correlation name (*ii...i*) contained in the SQL statement issued to the local HiRDB. If the correlation name is not used, neither (*hh...h*) nor (*ii...i*) is displayed.

If there is more than one table, each set of information is separated by a comma.

If the table identifiers are the same in both SQL statements, the correspondence is not displayed (same applies to the correlation names).

jj...j`[`*kk...k*`]`, . . .

Displays the correspondence between the variable *jj...j* contained in the SQL statement issued to the foreign server and the embedded variable, `?` parameter, SQL variable, SQL parameter, `USER`, `CURRENT_DATE`, `CURRENT_TIME`, or `CURRENT_TIMESTAMP`, or column specification *kk...k* contained in the SQL statement issued to the local HiRDB.

If there is more than one variable, each set of information is separated by a comma.

If the foreign DBMS is ORACLE, `:`*?-number* is displayed for variable *jj...j*; otherwise, `?(`*?-number*`)` is displayed for variable *jj...j* (*?-number*: sequential number assigned to `:`*?-number*, which is in the SQL statement issued to the foreign server, from left to right beginning at 1).

For *kk...k*, the actual value specified in the SQL statement is usually displayed as is, but in the case of an embedded variable or `?` parameter, `?(`*?-number*`)` is displayed (*?-number*: sequential number assigned to the embedded variable or `?` parameter, which is in the SQL statement issued to the local HiRDB, from left to right beginning at 1).

`SERVER-`*ll...l*`,COMM-`*mm...m*

Displays the predicted cost for the access to the foreign server.

ll...l

Displays the predicated cost for foreign server internal processing.

mm...m

Displays the predicted cost for communication from the foreign server to the local HiRDB.

17.5.2 Set operation information

```
----- QUERY EXPRESSION BODY ID : aa...a -----
Query Exp Type : bb...b cc...c(cc...c)  dd...d{AA...A}
Sub Query Type : ee...e{BB...B} [ff...f]
Transfer Type  : gg...g-CLM hh...h
Order by Mode  : ii...i
FLT Server     : jj...j
SetOpe Process : kk...k = [ll...l mm...m{CC...C}] nn...n [ll...l mm...m{CC...C}]
                    :
```

Explanation

aa...a

Displays the ID of the query expression.

The utility assigns a new number to each query expression including Set Operation. If the compiled SQL statement consists of multiple query expressions, information is separated by this line.

bb...b

Displays the type of query expression:

`QUERY`

Query

`DERIVED TABLE`

Derived table query when a query expression was specified in the `FROM` clause

`SUBQUERY`

Subquery

`WITH`

`WITH` clause query

`VIEW`

View definition query

`FOREIGN`

Query for accessing the foreign server

cc...c(*cc...c*)

One of the following, depending on the type of query expression:

- Name of the `WITH` clause query (correlation name) for a `WITH` clause query expression
- Name of the view table (correlation name) for a view definition query expression
- `(NO NAME)` or `(NO NAME)(`*correlation-name*`)` for a query expression in the derived table that was specified in the `FROM` clause

dd...d`{`*AA...A*`}`

Displays information when a work table is to be created for the set operation result. This information is not displayed if no work table is created.

If work table creation does not accompany a sort operation, the utility displays `LIST`.

If work table creation accompanies a sort operation, the utility displays `LIST(SORT)`.

ee...e

Displays the subquery execution method (for details, see *Execution of subqueries with no external reference* and *Execution of subqueries with external reference* in the *HiRDB Version 8 UAP Development Guide*):

`WORK TABLE ATS SUBQ`

The subquery execution method is work table ATS.

`WORK TABLE SUBQ`

The subquery execution method is work table.

`ROW VALUE SUBQ`

The subquery execution method is row value.

`HASH SUBQ`

The subquery execution method is hash.

`NESTED LOOPS WORK TABLE SUBQ`

The subquery execution method is nested loop work table.

`NESTED LOOPS ROW VALUE SUBQ`

The subquery execution method is nested loop row value.

ff...f

When the SQL optimization specification is used for the subquery execution method, displays whether or not the specification took effect (for details, see the manual *HiRDB Version 8 SQL Reference*):

`AS SPECIFIED`

SQL optimization specification was applied because the specification took effect.

`SPECIFICATION IGNORED`

SQL optimization specification was ignored because the specification did not take effect.

gg...g-`CLM` *hh...h*

Displays the method for transferring the result to the high-order query for a subquery, `WITH` clause query expression, view definition, or query expression created internally by HiRDB.

gg...g displays the number of rows used for transfer, and *hh...h* displays the type of transfer method. For details about the transfer method types, see *17.5.8 Types of transfer methods*.

ii...i

Displays the `ORDER BY` processing method for the set operation. When `ORDER BY` is not specified, this line is not displayed. For details about the `ORDER BY` processing method, see *17.5.14 Types of ORDER BY processing methods*.

For a HiRDB/Single Server, the `ORDER BY` processing method for queries is also displayed as query processing information.

jj...j

Displays the number of floating servers requested for the set operation. If no floating server is needed, `0` is displayed. This item is not displayed for a HiRDB/Single Server.

kk...k `=` `[`*ll...l* *mm...m*`{`*CC...C*`}]` *nn...n* `[`*ll...l* *mm...m*`{`*CC...C*`}]`

Displays the order of set operations. When multiple set operations are specified, this item consists of multiple lines.

kk...k

Displays the set operation number in the set operation results, in the format `"LID(`*set-operation-number*`)"`.

ll...l

Displays `"QID(`*query-ID*`)"` if the query expression to be joined is a query specification. If a join result of multiple query specifications is the query expression to be joined, this item displays `"LID(`*set-operation-number*`)"`.

mm...m`{`*CC...C*`}`

Displays information to create a work table before a set operation is executed. If no work table is created, this item is not displayed.

If a work table is created without sorting, `LIST` is displayed; if a work table is created with sorting, `LIST(SORT)` is displayed.

If the `LIMIT` clause is specified and only the first *n* items are obtained from the sorting results in memory without creating a work table, `MEM(SUBSORT)` is displayed.

nn...n

Displays the type of set operation ("`UNION`", "`UNION ALL`", "`EXCEPT`", or "`EXCEPT ALL`"). `[`*ll...l mm...m*`{`*CC...C*`}]` before and after *nn...n* indicate the query expressions to be joined.

The following is a display example of the order of set operations:

```
((SELECT C1 FROM T1    ...................query-ID:1
    UNION SELECT C1 FROM T2)    ..........query-ID:2
      UNION ALL SELECT C1 FROM T3)    ....query-ID:3
```

```
SetOpe Process : LID(1) = [QID(1) LIST(SORT) {...}] UNION [QID(2) LIST(SORT) {...}]
                 LID(2) = [LID(1) LIST{...}] UNION ALL [QID(3) LIST{...}]
```

17.5.3 Query processing information

```
----- QUERY ID : aa...a -----
Query Type     : bb...b  cc...c(cc...c)  dd...d{AA...A},...
Sub Query Type : ee...e{BB...B}  [ff...f]
IfThenCnd      :{CC...C}
Transfer Type  : gg...g-CLM  hh...h
Order by Mode  : ii...i
Having         :{DD...D}
Group by Mode  : jj...j
FLT Server     : kk...k(ll...l)  mm...m-CLM  nn...n
```

Explanation

aa...a

Displays the query ID.

The utility assigns a number to each query specification. If the compiled SQL query consists of multiple query specifications, the information is separated by this line.

For a query that includes access to a foreign table, the query ID for foreign server retrieval processing (prediction) information is the same as the query ID that is assigned to the SQL statement issued to the local HiRDB.

bb...b

Displays the type of query.

`QUERY`

Query

`DERIVED TABLE`

Derived table query (if a query expression was specified in the `FROM` clause)

`SUBQUERY`

Subquery

`WITH`

`WITH` clause query

`VIEW`

View definition query

`DUMMY`

Query internally created by HiRDB

`FOREIGN`

Query for accessing the foreign server

`CONSTRAINT`

Subquery in the referential constraint conditions

cc...c `(`*cc...c*`)`

Displays the following, depending on the query type:

- Name of the `WITH` clause query (correlation name) for a `WITH` clause query specification
- Name of the view table (correlation name) for a view definition query expression
- `(NO NAME)` or `(NO NAME)(`*correlation-name*`)` for a query expression in the derived table that was specified in the `FROM` clause
- If HiRDB created a query internally, the utility displays `(DUMMY` *work-table-number*`)` as the name of the temporary work table created internally to store the query results. The work table number is a 3-digit

integer.

- Name of the constraint for a subquery in the referential constraint conditions.

dd...d{*AA...A*}

Displays information if a work table is to be created to retrieve data and execute `GROUP BY`, `ORDER BY`, `DISTINCT`, set operation, or lock processing, etc., on the join result as required. This information is not displayed if no work table is created.

If multiple work tables are created due to a combination of operations, the tables are separated by the comma.

If work table creation does not accompany sorting, the utility displays `LIST`; if work table creation accompanies sorting, the utility displays `LIST(SORT)`.

When the `LIMIT` clause is specified, if the first *n* items in the sorting result are to be obtained in memory without creating a work table, the utility displays `MEM(SUBSORT)`.

ee...e{BB...B}

Displays the subquery execution method. For details about the subquery execution method, see the *HiRDB Version 8 UAP Development Guide*.

`WORK TABLE ATS SUBQ`

The subquery execution method is work table ATS.

`WORK TABLE SUBQ`

The subquery execution method is work table.

`ROW VALUE SUBQ`

The subquery execution method is row value.

`HASH SUBQ`

The subquery execution method is hash.

`NESTED LOOPS WORK TABLE SUBQ`

The subquery execution method is nested loop work table.

`NESTED LOOPS ROW VALUE SUBQ`

The subquery execution method is nested loop row value.

ff...f

If the SQL optimization specification is used for the subquery execution method, this item displays whether or not the specification takes effect. For details about the SQL optimization specification, see the manual *HiRDB Version 8 SQL*

Reference.

AS SPECIFIED: Takes effect

SPECIFICATION IGNORED: Ignored

gg...g-CLM hh...h

Displays the method for transferring the result to the high-order query for a subquery, WITH clause query, view definition query, set operation, or query created internally by HiRDB. *gg...g* displays the number of rows used for transfer, and *hh...h* displays the type of transfer method. For details about the types of transfer methods, see *17.5.8 Types of transfer methods*.

ii...i

Displays the ORDER BY processing method. The utility may execute ORDER BY processing implicitly even if the ORDER BY clause is not specified. For details about the ORDER BY processing method, see *17.5.14 Types of ORDER BY processing methods*.

If ORDER BY processing does not take place, the utility does not display this line.

jj...j

Displays the grouping method (including implicit grouping). For details about the grouping methods, see *17.5.10 Types of grouping processing methods*.

If grouping does not take place, the utility does not display this line.

kk...k

Displays the number of floating servers requested.

ll...l

Displays the purpose of using the floating servers.

During the retrieval, the utility uses all available floating servers within the number requested. This information is not displayed for a HiRDB/Single Server.

IMPLICIT GROUP BY

Purpose is to execute internal grouping for set function processing even if GROUP BY is not specified implicitly.

GROUP BY

Purpose is to execute grouping specified with GROUP BY.

DISTINCT

Purpose is to execute DISTINCT processing.

ORDER BY

Purpose is to execute `ORDER BY` processing.

`FOR READ ONLY`

Purpose is to process the `FOR READ ONLY` clause.

`FOR UPDATE`

Purpose is to process the `FOR UPDATE` clause.

`FOR UPDATE OF`

Purpose is to process `FOR UPDATE OF` *column-name*`[,`*column-name*`]...`

`PRE-UPDATE DATA`

Purpose is to allocate a floating server in order to reference pre-update data.

mm...m-CLM nn...n

Displays the method for transferring to the floating server. *mm...m* displays the number of rows used for transfer, and *nn...n* displays the type of transfer method. For details about the types of transfer methods, see *17.5.8 Types of transfer methods*.

17.5.4 Join processing information

(1) Join retrieval executed on a HiRDB/Single Server or HiRDB/Parallel Server (SELECT-APSL not used)

```
JOIN
 # Join ID      :aa...a
   L Join ID    : bb...b ff...f{BB...B}
   L Table      : cc...c(cc...c) dddddddddd(dd...d) ff...f{BB...B} {AA...A}
   L SubQ ID    : ee...e ff...f{BB...B}
   R Join ID    : gg...g kk...k{DD...D}
   R Table      : hh...h(hh...h) iiiiiiiii(ii...i) kk...k{DD...D} {CC...C}
   R SubQ ID    : jj...j kk...k{DD...D}
   R SubQEX ID  : tt...t kk...k{DD...D}
   JoinCnd      :{EE...E}
   Join Type    :ll...l-CLM mm...m(nn...n) oo...o-CLM pp...p(qq...q)  [rr...r]
   FLT Server   :ss...s
   RowCnd       :{FF...F}
   RowCndAfter  :{GG...G}
   IfThenCndOn  :{HH...H}
   IfThenCnd    :{II...I}
```

Explanation

aa...a

Displays the join processing ID.

The utility assigns this number to each join processing. If join processing is executed more than once, information about each join processing is separated by this line.

bb...b

If the left-hand term is another join result, the utility displays the corresponding join processing ID.

If the left-hand term is not a join result, the utility does not display this line.

cc...c (*cc...c*)

If the left-hand term is a table, the utility displays the table name (correlation name). The correlation name is not displayed if it is not used. Depending on the type of table, the following information is displayed:

- Name of the query for a derived table in the `WITH` clause
- Name of the view table (correlation name) for a view table
- `(NO NAME)` or `(NO NAME)(`*correlation-name*`)` for a derived table that was specified in the `FROM` clause

If the join target is not a table, this line is not displayed.

dddddddddd (*dd...d*)

If the left-hand term is a table, the utility displays the table ID in hexadecimal and in decimal (the latter in parentheses).

If the left-hand term is not a table, the utility does not display this line.

ee...e

If the left-hand join target is a subquery, the utility displays the subquery ID.

If the join target is not a subquery, this line is not displayed.

ff...f{*BB...B*}

Displays information if a work table is to be created before join processing. This information is not displayed if no work table is created.

If work table creation does not accompany a sort operation, the utility displays `LIST`. If work table creation accompanies a sort operation, the utility displays `LIST(SORT)`.

gg...g

If the right-hand term is another join result, the utility displays the corresponding join processing ID.

If the right-hand term is not a join result, the utility does not display this line.

hh...h(*hh...h*)

If the right-hand term is a table, the utility displays the table name (correlation name).The correlation name is not displayed if it is not used. Depending on the type of table, the following information is displayed:

- Name of the query for a derived table in the `WITH` clause
- Name of the view table (correlation name) for a view table
- `(NO NAME)` or `(NO NAME)(`*correlation-name*`)` for a derived table that was specified in the `FROM` clause

If the right-hand term is not a table, the utility does not display this line.

iiiiiiiii(*ii...i*)

If the right-hand term is a table, the utility displays the table ID in hexadecimal and in decimal (the latter in parentheses).

If the right-hand term is not a table, the utility does not display this line.

jj...j

If the right-hand join target is a subquery that includes no set operation, the utility displays the subquery ID. If the join target is not a subquery that includes no set operation, this line is not displayed.

kk...k{*DD...D*}

Displays information if a work table is to be created before executing join processing. This information is not displayed if no work table is created.

If work table creation does not accompany a sort operation, the utility displays `LIST`. If work table creation accompanies a sort operation, the utility displays `LIST(SORT)`.

ll...l-`CLM` *mm...m*(*nn...n*)

Displays the join method:

ll...l

Displays the number of join conditions (number of columns) used for join processing.

mm...m

Displays the type of join method. For details about types of join methods, see *17.5.6 Types of join methods*.

nn...n

Displays the execution type of join processing:

- `INNER`

  Inner join. The utility evaluates the retrieval condition and returns the join result if the condition is true.

- `LEFT OUTER`

  Left outer join. The utility evaluates the `ON` retrieval condition and returns the join result if the condition is true. If none of the inner table's rows is true for the outer table's row, the utility fills the null value in the inner table's column and returns the join result.

- `EXIST`

  The utility evaluates the retrieval condition and returns the outer table's row if at least one of the rows in the inner table is true for the outer table's row.

- `NOT EXIST`

  The utility evaluates the retrieval condition and returns the outer table's row if none of the rows in the inner table is true for the outer table's row.

- `ALL`

  The utility evaluates the retrieval condition and returns the outer table's row if all rows in the inner table are true for the outer table's row.

- `VALUE`

  The utility evaluates the retrieval condition and returns the outer table's row if only one of the rows in the inner table is true for the outer table's row. If more than one row in the inner table is true, an error results.

oo...o`-CLM` *pp...p*`(`*qq...q*`)`

Displays the method for transferring to the server used for join processing:

oo...o

Displays the number of columns used for transfer.

pp...p

Displays the type of transfer method. For details about types of transfer methods, see *17.5.8 Types of transfer methods*.

qq...q

Displays the transfer direction:

- `FROM L TO R`

  Transfers from left to right to achieve join processing.

- `FROM R TO L`

  Transfers from right to left to achieve join processing.

- `TO FLT`

  Transfers both left- and right-hand terms to the floating server to achieve join processing.

rr...r

If the SQL optimization specification is used for the join method, this item displays whether or not the specification takes effect. For details about the SQL optimization specification, see the manual *HiRDB Version 8 SQL Reference*.

`AS SPECIFIED`: Takes effect

`SPECIFICATION IGNORED`: Ignored

ss...s

Displays the number of floating servers requested for sort processing on the HiRDB/Parallel Server. During the retrieval, the utility uses all available floating servers within the number requested. If the join method is `NESTED LOOPS JOIN`, this value is 0. This information is not displayed for a HiRDB/Single Server.

tt...t

If the right-hand join target is a subquery that includes a set operation, the utility displays the subquery ID.

If the join target is not a subquery containing a set operation, this line is not displayed.

(2) Join retrieval executed on a HiRDB/Parallel Server (SELECT-APSL used)

```
JOIN
 # Join ID      :aa...a
   L Join ID    : bb...b  ee...e{AA...A}
   L Table      : cc...c(cc...c)  dddddddddd(dd...d)    ee...e{AA...A}
   R Join ID    : ff...f   ii...i{BB...B}
   R Table      : gg...g(gg...g)  hhhhhhhhhh(hh...h)    ii...i{BB...B}
   JoinCnd      :{CC...C}
   Join Type    :SELECT-APSL
         Table Name    :jj...j(jj...j)  kkkkkkkkkk(kk...k)
         Column ID     :llllllllll(ll...l)
         Predicate     :mm...m
         Threshold     :nn...n
      [1] oo...o-CLM pp...p(qq...q)   rr...r-CLM ss...s(tt...t)  ........1
          FLT Server  :uu...u
          IfThenCnd   :{DD...D}
      [2] oo...o-CLM pp...p(qq...q)   rr...r-CLM ss...s(tt...t)  ........2
          FLT Server  :uu...u
          IfThenCnd   :{DD...D}
```

Explanation

1 and *2* indicate the first candidate and second candidate, respectively, for the join method.

aa...a

Displays the join processing ID.

The utility assigns this number to each join processing. If join processing is executed more than once, information about each join processing is separated by this line.

bb...b

If the left-hand term is another join result, the utility displays the corresponding join processing ID.

If the left-hand term is not a join result, the utility does not display this line.

cc...c (*cc...c*)

If the left-hand term is a table, the utility displays the table name (correlation name).

The correlation name is not displayed if it is not used. Depending on the type of table, the following information is displayed:

- Name of the query for a derived table in the `WITH` clause
- Name of the view table (correlation name) for a view table

- `(NO NAME)` or `(NO NAME)(`*correlation-name*`)` for a derived table that was specified in the `FROM` clause

If the left-hand term is not a table, the utility does not display this line.

dddddddddd`(`*dd...d*`)`

If the left-hand term is a table, the utility displays the table ID in hexadecimal and in decimal (the latter in parentheses).

If the left-hand term is not a table, the utility does not display this line.

ee...e`{`*AA...A*`}`

Displays information if a work table is to be created before join processing. This information is not displayed if no work table is created.

If work table creation does not accompany a sort operation, the utility displays `LIST`. If work table creation accompanies a sort operation, the utility displays `LIST(SORT).`

ff...f

If the right-hand term is another join result, the utility displays the corresponding join processing ID.

If the right-hand term is not a join result, the utility does not display this line.

gg...g`(`*gg...g*`)`

If the right-hand term is a table, the utility displays the table name (correlation name).

The correlation name is not displayed if it is not used. Depending on the type of table, the following information is displayed:

- Name of the query for a derived table in the `WITH` clause
- Name of the view table (correlation name) for a view table
- `(NO NAME)` or `(NO NAME)(`*correlation-name*`)` for a derived table that was specified in the `FROM` clause

If the right-hand term is not a table, the utility does not display this line.

hhhhhhhhhh`(`*hh...h*`)`

If the right-hand term is a table, displays the table ID in hexadecimal and in decimal (the latter in parentheses).

If the right-hand term is not a table, the utility does not display this line.

ii...i`{`*BB...B*`}`

Displays information if a work table is to be created before join processing. This information is not displayed if no work table is created.

If work table creation does not accompany a sort operation, the utility displays `LIST`. If work table creation accompanies a sort operation, the utility displays `LIST(SORT)`.

jj...j (*jj...j*)

Displays the table name (correlation name) for the column used in the predicate that is subject to calculation of the SQL run-time hit rate when selecting an SQL object.

kkkkkkkkkk (*kk...k*)

Displays in hexadecimal and in decimal (the latter in parentheses) the table ID for the column used in the predicate that is subject to calculation of the SQL run-time hit rate when selecting an SQL object.

llllllllll (*ll...l*)

Displays in hexadecimal and in decimal (the latter in parentheses) the column ID used in the predicate that is subject to calculation of the SQL run-time hit rate when selecting an SQL object.

mm...m

Displays the predicate subject to calculation of the SQL run-time hit rate when selecting an SQL object.

nn...n

Displays the reference hit rate for selecting an SQL object.

If the hit rate obtained from calculation is less than this value, the first candidate is used for the SQL object; if it is equal to or greater than this value, the second candidate is used for the SQL object.

oo...o-`CLM` *pp...p* (*qq...q*)

Displays the join method used:

oo...o

Displays the number of join conditions (number of columns) used for join processing.

pp...p

Displays the type of join method. For details about types of join methods, see *17.5.6 Types of join methods*.

qq...q

Displays the execution type of join processing:

`INNER`: Inner join. The utility evaluates the retrieval condition and returns the join result if the condition is true.

rr...r-`CLM` *ss...s*`(`*tt...t*`)`

Displays the method for transferring to the server used for join processing:

rr...r

Displays the number of columns used for transfer.

ss...s

Displays the type of transfer method. For details about types of transfer methods, see *17.5.8 Types of transfer methods*.

tt...t

Displays the transfer direction:

- `FROM L TO R`

  Transfers from left to right to achieve join processing.

- `FROM R TO L`

  Transfers from right to left to achieve join processing.

- `TO FLT`

  Transfers both left- and right-hand terms to the floating server to achieve join processing.

uu...u

Displays the number of floating servers requested for sort processing on the HiRDB/Parallel Server. During the retrieval, the utility uses all available floating servers within the number requested. If the join method is `NESTED LOOPS JOIN`, this value is 0. This information is not displayed for a HiRDB/Single Server.

17.5.5 Base table retrieval processing information

(1) Base table retrieval executed on a HiRDB/Single Server or HiRDB/Parallel Server (SELECT-APSL not used) without using an index or using only one index

```
SCAN
 # Table Name  :aa...a(aa...a) bbbbbbbbbb(bb...b) (xx...x) cc...c{AA...A}
   Cost        :d(ee...eROW) {T-BB...B,I-CC...C,P-DD...D,AND-EE...E,OR-FF...F}
   RDAREA      :ff...f-CLM gg...g(hh...h) (ii...iRD/jj...jBES) [kkkk(kk...k),...] ll...l
   Rebalance   :mm...m
   Scan Type   :nn...n(oo...o)  [pp...p]
   Index Name  :qq...q rrrrrrrrrr(rr...r) (ss...s)tt...t
                     SearchCnd :uu...u[vv...v],...
                     KeyCnd    :ww...w
   RowCnd      :{HH...H}
   IfThenCnd   :{II...I}
```

Explanation

The RDAREA row is not displayed for retrieval processing information for a foreign table.

If the INSERT statement with VALUES specified was executed, the Scan Type line is not displayed.

aa...a(*aa...a*)

Displays the name of the table (correlation name) being retrieved.

The correlation name is not displayed if it is not used. If retrieval processing is executed more than once, information about each retrieval processing is separated by this line.

bbbbbbbbbb(*bb...b*)

Displays in hexadecimal and in decimal (the latter in parentheses) the ID of the table being retrieved.

cc...c{*AA...A*}

Displays information if a work table is to be created after data retrieval from the base table. This information is not displayed if no work table is created.

If work table creation does not accompany a sort operation, the utility displays LIST. If work table creation accompanies a sort operation, the utility displays LIST(SORT).

d

Displays whether or not optimization information is to be collected by the optimizing information collection utility (pdgetcst):

`Y`: Optimization information collected

`N`: Optimization information not collected

ee...e

Displays the number of table rows used by HiRDB to determine the access path.

The default value depends on the SQL optimization mode:

- Cost-based optimization mode 1 or cost-based optimization mode 2 with a HiRDB/Single Server

  10 million lines
- Cost-based optimization mode 1 with a HiRDB/Parallel Server

  10 million lines × number of back-end servers containing the table storage RDAREAs

By executing the optimizing information collection utility (`pdgetcst`), you can change the number of table rows to be used by HiRDB to determine the access path.

ff...f

Displays the number of columns into which a table is divided.

gg...g

Displays the type of table partitioning:

`NON DIVISION`: not partitioned

`KEY RANGE`: key range partitioning (with storage conditions specified)

`PARTITION`: key range partitioning (with boundary values specified)

`MULTIDIM PARTITION`: matrix partitioning (with boundary values specified)

`MULTIDIM`: Matrix partitioning including hash partitioning

`FLEXIBLE HASH`: flexible hash partitioning without using the rebalancing facility

`FIX HASH`: fix hash partitioning without using the rebalancing facility

`RB FLEXIBLE HASH`: flexible hash partitioning using the rebalancing facility

`RB FIX HASH`: fix hash partitioning using the rebalancing facility

hh...h

Displays the hash function during hash partitioning. This information is not displayed when hash partitioning is not performed. This information is not displayed for matrix partitioning that includes hash partitioning.

`(`*ii...i*`RD/`*jj...j*`BES)`

ii...i displays the number of table partitions (number of RDAREAs) and *jj...j* displays the number of back-end servers containing these table storage RDAREAs.

For a non-partitioned table, *ii...i* displays a value of 1. /*jj...j*`BES` is not displayed for a HiRDB/Single Server.

kkkk`(`*kk...k*`)`

Displays the ID of the table storage RDAREA in hexadecimal and in decimal (the latter in parentheses).

If the table is partitioned, the utility displays as many RDAREA IDs as there are table partitions. To obtain the name of the RDAREA associated with a given RDAREA ID, execute the `pddbls` command.

ll...l

`ALL`

This is displayed when all RDAREAs defining a table are to be retrieved.

`RESTRICTED`

This is displayed when, among the RDAREAs defining the table, only those RDAREAs that can potentially contain the target data based on the specified search conditions are to be retrieved.

mm...m

Displays the rebalancing status for a table using the rebalancing facility.

This line is not displayed for a table that does not use the rebalancing facility. For details about the rebalancing facility, see the *HiRDB Version 8 System Operation Guide*.

`NORMAL`

This is the normal status or the status when `ALTER TABLE` was used to add an RDAREA but the rebalancing utility (`pdrbal`) has not been executed yet.

`ON REBALANCE`

This is the status when the table is being rebalanced (from the start to the end of rebalance operation).

nn...n

Displays the retrieval method to be used.

For details about types of retrieval methods, see *17.5.7 Types of retrieval methods*.

oo...o

Displays the index retrieval method if `Group by Mode` is `IMPLICIT MIN-MAX`

`INDEX`

`MIN`

The utility searches a retrieval range of the index in ascending order and stops the retrieval at the point where a row satisfying the retrieval condition is found.

`MAX`

The utility searches a retrieval range of the index in descending order and stops the retrieval at the point where a row satisfying the retrieval condition is found.

`MIN,MAX`

The utility first searches the index in ascending order to find the `MIN` value, then searches the index in descending order to find the `MAX` value.

The retrieval time is about twice as much as when either `MIN` or `MAX` is specified in the set function.

Because `MIN` and `MAX` use mutually opposite search directions, their index search ranges based on the search condition are also opposite. If both `MIN` and `MAX` are used, the utility displays the search range only or `MIN`.

pp...p

If the SQL optimization specification is used for the index used, this item displays whether or not the specification takes effect. For details about the SQL optimization specification, see the manual *HiRDB Version 8 SQL Reference*.

`AS SPECIFIED`: Takes effect

`SPECIFICATION IGNORED`: Ignored

`PARTIALLY IGNORED`: Part of the specification is ignored

qq...q

Displays the name of the index used for retrieval.

If no index is used for retrieval, the utility does not display this line.

rrrrrrrrrr(rr...r)

Displays the index ID used for retrieval in hexadecimal and in decimal (the latter in parentheses).

ss...s

Displays a combination of index attributes:

Numeric character: Number of columns composing the index

`G`: Plug-in index

`U`: Unique key index

`C`: Cluster key index

`D`: Partition key index or plug-in index

`d`: Non-partitioning key index partitioned in the server

`E`: Index including an exceptional key value

`M`: Index containing repetition columns

`P`: Primary key index

tt...t

Indexes other than the plug-in index:

The utility displays the index component column information by separating each column by a comma and enclosing the entire information item in parentheses.

The index component column information consists of the search direction and the name of the index component column.

As the search direction, + means that the component column is searched in ascending order while - means that it is searched in descending order.

Example

Searching component column 1 (`C1`) in ascending order and component column 2 (`C2`) in descending order:

```
(+C1,-C2)
```

Plug-in index:

The utility displays detailed information according to types of plug-in-provided functions. For details about the types of plug-in-provided functions, see *17.5.11 Types of plug-in-provided functions*. For the function call, only the first argument is displayed.

Example 1

Searching component column 1 (`C1`) using a plug-in-provided function (`WITHIN`) with type `SCAN TYPE`:

```
WITHIN(C1,..)[SCAN TYPE]
```

Example 2

Searching component column 1 (`C1`) using a plug-in-provided function (`CONTAINS`) with type `INDEX SCAN TYPE`:

```
CONTAINS (C1,..)[INDEX SCAN TYPE]
```

Example 3

Searching component column 1 (`C1`) using a plug-in-provided function (`SEARCHFEATUREDATA`) with type `FULL SCAN TYPE`:
`SEARCHFEATUREDATA (C1,..)[FULL SCAN TYPE]`

uu...u`[`*vv...v*`],..`

Displays the type and narrowed range of search condition. For details about the search conditions, see *17.5.12 Search conditions*.

If there is no search condition for a retrieval using an index, the utility displays `NONE(FULL SCAN)` during the index all-range search. If the `Group by mode` is `IMPLICIT MIN-MAX INDEX`, the utility displays `NONE` (when the search condition becomes true, the utility stops the search, in which case `(FULL SCAN)` is not displayed because this may not be the index all-range search).

The number of narrowed ranges may become `0` or `1` depending on the search condition. If there is no search condition for the first component column, but there are search conditions for the subsequent columns, the utility displays `(FULL SCAN)` following the narrowed range of search conditions.

ww...w

Displays the key condition. For details about key conditions, see *17.5.13 Key conditions*.

If there is no key condition, the utility does not display this line.

xx...x

If a shared table is the search target, the utility displays `SHARED`; if not, the utility does not display this item.

(2) Base table retrieval executed on a HiRDB/Single Server or a HiRDB/Parallel Server (SELECT-APSL not used) using at least two indexes

```
SCAN
 # Table Name    :aa...a(aa...a) bbbbbbbbbb(bb...b) (CC...C) cc...c{DD...DD}
   Cost          :d (ee...eROW)  {T-EE...E,I-FF...F,P-GG...G,AND-HH...H,OR-II...I}
   RDAREA        :ff..f-CLM gg...g(hh...h) (ii...iRD/jj...jBES) [kkkk(kk...k),...] LL...L
   Rebalance     :mm...m
   Scan Type     :nn...n  [oo...o]
   Index Name    :pp...p = qq...q rrrrrrrrrr(rr...r) (ss...s) tt...t BB...B{KK...K}
                          Scan Type :uu...u
                          SearchCnd :vv...v[ww...w],...
                          KeyCnd    :xx...x
                          RowCnd    :{HH...H}
                          IfThenCnd :{NN...N}
                  pp...p = qq...q rrrrrrrrrr(rr...r) (ss...s) tt...t BB...B{KK...K}
                          Scan Type :uu...u
                          SearchCnd :vv...v[ww...w],...
                          KeyCnd    :xx...x
                          RowCnd    :{HH...H}
                          IfThenCnd :{NN...N}
                  pp...p = [yy...y zz...z{JJ...J}] AA...A[yy...y zz...z{JJ...J}]
   RowCnd        :{MM...M}
   IfThenCnd     :{OO...O}
```

Explanation

The `RDAREA` row is not displayed for retrieval processing information for a foreign table.

aa...a (*aa...a*)

Displays the name of the table (correlation name) being retrieved.

The correlation name is not displayed if it is not used. If retrieval processing is executed more than once, information about each retrieval processing is separated by this line.

bbbbbbbbbb (*bb...b*)

Displays in hexadecimal and in decimal (the latter in parentheses) the ID of the table being retrieved.

cc...c {*DD...D*}

Displays information if a work table is to be created after data retrieval from the base table. This information is not displayed if no work table is created.

If work table creation does not accompany a sort operation, the utility displays `LIST`. If work table creation accompanies a sort operation, the utility displays `LIST(SORT)`.

d

Displays whether or not optimization information is to be collected by the optimizing information collection utility (`pdgetcst`):

`Y`: Optimization information collected

`N`: Optimization information not collected

ee...e

Displays the number of table rows used by HiRDB to determine the access path.

The default value depends on the SQL optimization mode:

- Cost-based optimization mode 1 or cost-based optimization mode 2 with a HiRDB/Single Server

  10 million lines

- Cost-based optimization mode 1 with a HiRDB/Parallel Server

  10 million lines × number of back-end servers containing the table storage RDAREAs

By executing the optimizing information collection utility (`pdgetcst`), you can change the number of table rows to be used by HiRDB to determine the access path.

ff...f

Displays the number of columns into which a table is divided.

gg...g

Displays the type of table partitioning:

`NON DIVISION`: not partitioned

`KEY RANGE`: key range partitioning (with storage conditions specified)

`PARTITION`: key range partitioning (with boundary values specified)

`MULTIDIM PARTITION`: matrix partitioning (with boundary values specified)

`MULTIDIM`: Matrix partitioning including hash partitioning

`FLEXIBLE HASH`: flexible hash partitioning without using the rebalancing facility

`FIX HASH`: fix hash partitioning without using the rebalancing facility

`RB FLEXIBLE HASH`: flexible hash partitioning using the rebalancing facility

`RB FIX HASH`: fix hash partitioning using the rebalancing facility

hh...h

Displays the hash function during hash partitioning. This information is not displayed when hash partitioning is not performed. This information is not displayed for matrix partitioning that includes hash partitioning.

`(`*ii...i*`RD/`*jj...j*`BES)`

ii...i displays the number of table partitions (number of RDAREAs) and *jj...j* displays the number of back-end servers containing these table storage RDAREAs.

For a non-partitioned table, *ii...i* displays a value of 1. `/`*jj...j*`BES` is not displayed for a HiRDB/Single Server.

kkkk`(`*kk...k*`)`

Displays the ID of the table storage RDAREA in hexadecimal and in decimal (the latter in parentheses).

If the table is partitioned, the utility displays as many RDAREA IDs as there are table partitions. To obtain the name of the RDAREA associated with a given RDAREA ID, execute the `pddbls` command.

ll...l

`ALL`

This is displayed when all RDAREAs defining a table are to be retrieved.

`RESTRICTED`

This is displayed when, among the RDAREAs defining the table, only those RDAREAs that can potentially contain the target data based on the specified search conditions are to be retrieved.

mm...m

Displays the rebalancing status for a table using the rebalancing facility.

This line is not displayed for a table that does not use the rebalancing facility. For details about the rebalancing facility, see the *HiRDB Version 8 System Operation Guide*.

`NORMAL`

This is the normal status or the status when `ALTER TABLE` was used to add an RDAREA but the rebalancing utility (`pdrbal`) has not been executed yet.

`ON REBALANCE`

This is the status when the table is being rebalanced (from the start to the end of rebalance operation).

nn...n

Displays the retrieval method to be used.

For details about types of retrieval methods, see *17.5.7 Types of retrieval methods*.

oo...o

If the SQL optimization specification is used for the index used, this item displays whether or not the specification takes effect. For details about the SQL optimization specification, see the manual *HiRDB Version 8 SQL Reference*.

`AS SPECIFIED`: Takes effect

`SPECIFICATION IGNORED`: Ignored

`PARTIALLY IGNORED`: Part of the specification is ignored

pp...p

Displays the work table number to be created for `AND PLURAL INDEXES SCAN` in the format `LID(`*work-table-number*`)`.

qq...q

Displays the index name used to create a work table for `AND PLURAL INDEXES SCAN` or `OR PLURAL INDEXES SCAN` as many times as there are such indexes. For the work table created without using an index, the utility displays `(NO USE)`.

rrrrrrrrrr(rr...r)

Displays the index ID used for retrieval in hexadecimal and in decimal (the latter in parentheses).

ss...s

Displays a combination of index attributes:

Numeric character: Number of columns composing the index

`G`: Plug-in index

`U`: Unique key index

`C`: Cluster key index

`D`: Partition key index or plug-in index

`d`: Non-partitioning key index partitioned in the server

`E`: Index including an exceptional key value

`M`: Index containing repetition columns

`P`: Primary key index

tt...t

Indexes other than the plug-in index:

The utility displays the index component column information by separating each column by a comma and enclosing the entire information item in parentheses.

The index component column information consists of the search direction and the name of the index component column.

As the search direction, + means that the component column is searched in ascending order while - means that it is searched in descending order.

Example

Searching component column 1 (`C1`) in the ascending order and component column 2 (`C2`) in the descending order:

```
(+C1,-C2)
```

Plug-in index:

The utility displays detailed information according to types of plug-in-provided functions. For details about the types of plug-in-provided functions, see *17.5.11 Types of plug-in-provided functions*. For the function call, only the first argument is displayed.

Example 1

Searching component column 1 (`C1`) using a plug-in-provided function (`WITHIN`) with type `SCAN TYPE`:

```
WITHIN(C1,..)[SCAN TYPE]
```

Example 2

Searching component column 1 (`C1`) using a plug-in-provided function (`CONTAINS`) with type `INDEX SCAN TYPE`:

```
CONTAINS (C1,..)[INDEX SCAN TYPE]
```

Example 3

Searching component column 1 (`C1`) using a plug-in-provided function (`SEARCHFEATUREDATA`) with type `FULL SCAN TYPE`:

```
SEARCHFEATUREDATA (C1,..)[FULL SCAN TYPE]
```

uu...u

Displays the search method for creating each work table.

For details about the types of retrieval methods, see *17.5.7 Types of retrieval methods*.

vv...v`[`*ww...w*`],..`

Displays the type and narrowed range of search condition. For details about the search conditions, see *17.5.12 Search conditions*.

If there is no search condition for a retrieval using an index, the utility displays `NONE(FULL SCAN)` as the type of search condition.

The number of narrowed ranges may become 0 or 1 depending on the search condition. If there is no search condition for the first component column, but there are search conditions for the subsequent columns, the utility displays `(FULL SCAN)` following the narrowed range of search condition.

xx...x

Displays the key condition. For details about key conditions, see *17.5.13 Key conditions*.

If there is no key condition, the utility does not display this line.

pp...p = [*yy...y* *zz...z*{*JJ...J*}] *AA...A* [*yy...y* *zz...z*{*JJ...J*}]

Displays the order in which work tables are created during `AND PLURAL INDEXES SCAN`. If more than two indexes are used for retrieval, this information is displayed for each work table.

pp...p displays the work table number in the format `LID(`*work-table-number*`)`.

AA...A displays `AND`, `OR`, or `ANDNOT` as the type of operation performed on the work tables.

yy...y displays the input work table for the operation in the format `LID(`*work-table-number*`)`.

For *zz...z*, the utility displays `LIST` if work table creation does not accompany sorting; it displays `LIST(SORT)` if work table creation accompanies sorting.

Example: The following SQL statement was executed for a table with indexes `IX1(C1)` and `IX2(C2)` defined:

```
where C1='A'                        ← work table number: 1
  or C2=between 'a' and 'z'         ← work table number: 2
```

If `Scantype` is `AND PLURAL INDEXES SCAN`, then the following result is output:

```
    :
Scan Type     :AND PLURAL INDEXES SCAN
Index Name    :LID(1)=IX1 (1) (+C1)
                      Scan Type :INDEX SCAN
                      SearchCnd :AT['A']
               LID(2)=IX2 (1) (+C2)
                      Scan Type :INDEX SCAN
                      SearchCnd :RANGE(CS-CE) ['a','z']
               LID(3)=[LID(1) LIST(SORT){..}] OR [LID(2) LIST(SORT){..}]
```

BB...B{*KK...K*}

If a work table is to be created after the table search separately from the work table for `AND PLURAL INDEXES SCAN`, the utility displays this information.

If the work table is to be created without sorting, the utility displays `LIST` for *II...I*. If work table creation involves sorting, the utility displays `LIST(SORT)` for *II...I*. If the work table is not to be created, the utility does not display this information.

CC...C

If a shared table is the search target, the utility displays `SHARED`; if not, the utility does not display this item.

(3) Base table retrieval executed on a HiRDB/Parallel Server (SELECT-APSL used)

```
SCAN
 # Table Name    :aa...a(aa...a) bbbbbbbbbb(bb...b) (zz...z)
   Cost          :c (dd...dROW) {T-AA...A,I-BB...B,P-CC...C,AND-DD...D,OR-EE...E}
   RDAREA        :ee...e-CLM ff...f(gg...g) (hh...hRD/ii...iBES) [jjjj(jj...j),...] kk...k
   Rebalance     :ll...l
   Scan Type     :SELECT-APSL
         Table Name    :mm...m(mm...m) nnnnnnnnnn(nn...n)
         Column ID     :oooooo(oo...o)
         Predicate     :pp...p
         Threshold     :qq...q
      [1] rr...r  ................................................1
           Index Name :ss...s tttttttttt(tt...t) (uu...u) vv...v
                           SearchCnd :ww...w[xx...x],...
                           KeyCnd    :yy...y
                           RowCnd    :{FF...F}
      [2] rr...r  ................................................2
           Index Name :ss...s tttttttttt(tt...t) (uu...u) vv...v
                           SearchCnd :ww...w[xx...x],...
                           KeyCnd    :yy...y
                           RowCnd    :{FF...F}
   IfThenCnd     :{GG...G}
```

Explanation

Nos. 1 and 2 indicate the first and second candidates, respectively, for the retrieval method. For details about the retrieval method, see *17.5.7 Types of retrieval methods*.

The `RDAREA` row is not displayed for retrieval processing information for a foreign table.

aa...a(*aa...a*)

Displays the name of the table (correlation name) being retrieved.

The correlation name is not displayed if it is not used. If retrieval processing is executed more than once, information about each retrieval processing is separated by this line.

bbbbbbbbbb(*bb...b*)

Displays in hexadecimal and in decimal (the latter in parentheses) the ID of the table being retrieved.

c

Displays whether or not optimization information is to be collected by the optimizing information collection utility (`pdgetcst`):

`Y`: Optimization information collected

`N`: Optimization information not collected

dd...d

Displays the number of table rows used by HiRDB to determine the access path.

The default value depends on the SQL optimization mode:

- Cost-based optimization mode 1 or cost-based optimization mode 2 with a HiRDB/Single Server

  10 million lines
- Cost-based optimization mode 1 with a HiRDB/Parallel Server

  10 million lines × number of back-end servers containing the table storage RDAREAs

By executing the optimizing information collection utility (`pdgetcst`), you can change the number of table rows to be used by HiRDB to determine the access path.

ee...e

Displays the number of columns into which a table is divided.

ff...f

Displays the type of table partitioning:

`NON DIVISION`: not partitioned

`KEY RANGE`: key range partitioning (with storage conditions specified)

`PARTITION`: key range partitioning (with boundary values specified)

`MULTIDIM PARTITION`: matrix partitioning (with boundary values specified)

`MULTIDIM`: Matrix partitioning including hash partitioning

`FLEXIBLE HASH`: flexible hash partitioning without using the rebalancing facility

`FIX HASH`: fix hash partitioning without using the rebalancing facility

`RB FLEXIBLE HASH`: flexible hash partitioning using the rebalancing facility

`RB FIX HASH`: fix hash partitioning using the rebalancing facility

gg...g

Displays the hash function during hash partitioning. This information is not displayed when hash partitioning is not performed. This information is not displayed for matrix partitioning that includes hash partitioning.

(*hh...h*RD/*ii...i*BES)

hh...h displays the number of table partitions (number of RDAREAs) and *ii...i* displays the number of back-end servers containing these table storage RDAREAs.

For a non-partitioned table, *hh...h* displays a value of 1. /*ii...i*BES is not displayed for a HiRDB/Single Server.

jjjj (*jj...j*)

Displays the ID of the table storage RDAREA in hexadecimal and in decimal (the latter in parentheses).

If the table is partitioned, the utility displays as many RDAREA IDs as there are table partitions. To obtain the name of the RDAREA associated with a given RDAREA ID, execute the `pddbls` command.

kk...k

`ALL`

This is displayed when all RDAREAs defining a table are to be retrieved.

`RESTRICTED`

This is displayed when, among the RDAREAs defining the table, only those RDAREAs that can potentially contain the target data based on the specified search conditions are to be retrieved.

ll...l

Displays the rebalancing status for a table using the rebalancing facility.

This line is not displayed for a table that does not use the rebalancing facility. For details about the rebalancing facility, see the *HiRDB Version 8 System Operation Guide*.

`NORMAL`

This is the normal status or the status when `ALTER TABLE` was used to add an RDAREA but the rebalancing utility (`pdrbal`) has not been executed yet.

`ON REBALANCE`

This is the status when the table is being rebalanced (from the start to the end of rebalance operation).

mm...m (*mm...m*)

Displays the table name (correlation name) for the column used in the predicate that is subject to calculation of the SQL run-time hit rate when selecting an SQL object.

nnnnnnnnnn (*nn...n*)

Displays in hexadecimal and in decimal (the latter in parentheses) the table ID for the column used in the predicate that is subject to calculation of the SQL run-time hit rate when selecting an SQL object.

oooooo (*oo...o*)

Displays in hexadecimal and in decimal (the latter in parentheses) the column ID used in the predicate that is subject to calculation of the SQL run-time hit rate when selecting an SQL object.

pp...p

Displays the predicate subject to calculation of the SQL run-time hit rate when selecting an SQL object.

qq...q

Displays the reference hit rate for selecting an SQL object.

If the hit rate obtained from calculation is less than this value, the first candidate is used for the SQL object; if it is equal to or greater than this value, the second candidate is used for the SQL object.

rr...r

Displays the retrieval method to be used.

For details about the types of retrieval methods, see *17.5.7 Types of retrieval methods*.

ss...s

Displays the name of the index used for retrieval.

If no index is used for retrieval, the utility does not display this line.

ttttttttt (*tt...t*)

Displays the index ID used for retrieval in hexadecimal and in decimal (the latter in parentheses).

uu...u

Displays a combination of index attributes:

Numeric character: Number of columns composing the index

`G`: Plug-in index

`U`: Unique key index

`C`: Cluster key index

`D`: Partition key index or plug-in index

`d`: Non-partitioning key index partitioned in the server

`E`: Index including an exceptional key value

`M`: Index containing repetition columns

`P`: Primary key index

vv...v

The utility displays the index component column information by separating each column by a comma and enclosing the entire information item in parentheses.

The index component column information consists of the search direction and the name of the index component column.

As the search direction, `+` means that the component column is searched in ascending order while `-` means that it is searched in descending order.

Example

Searching component column 1 (`C1`) in ascending order and component column 2 (`C2`) in descending order:
```
(+C1,-C2)
```

ww...w`[`*xx...x*`],..`

Displays the type and narrowed range of search condition. For details about the search conditions, see *17.5.12 Search conditions*.

If there is no search condition for a retrieval using an index, the utility displays `NONE(FULL SCAN)` as the type of search condition.

The number of narrowed ranges may become 0 or 1 depending on the search condition. If there is no search condition for the first component column, but there are search conditions for the subsequent columns, the utility displays `(FULL SCAN)` following the narrowed range of search conditions.

yy...y

Displays the key condition. For details about key conditions, see *17.5.13 Key* *conditions*.

If there is no key condition, the utility does not display this line.

zz...z

If a shared table is the search target, the utility displays `SHARED`; if not, the utility does not display this item.

(4) Base table retrieval (SELECT-APSL used) combined with join processing (SELECT-APSL used) executed on a HiRDB/Parallel Server

```
SCAN
 # Table Name    :aa...a(aa...a) bbbbbbbbbb(bb...b) (AA...A)
   Cost          :c (dd...dROW) {T-BB...B,I-CC...C,P-DD...D,AND-EE...E,OR-FF...F}
   RDAREA        :ee...e-CLM ff...f(gg...g) (hh...hRD/ii...iBES) [jjjj(jj...j),...] kk...k
   Rebalance     :ll...l
   Scan Type     :SELECT-APSL
         Table Name    :mm...m(mm...m) nnnnnnnnnn(nn...n)
         Column ID     :oooooo(oo...o)
         Predicate     :pp...p
         Threshold     :qq...q
      [1] rr...r (ss...s) .......................................1
           Index Name :tt...t uuuuuuuuuu(uu...u) (vv...v) ww...w
                          SearchCnd :xx...x[yy...y],...
                          KeyCnd    :zz...z
                          RowCnd    :{GG...G}
                          IfThenCnd :{HH...H}
      [2] SELECT-APSL (ss...s) ..................................2
            Table Name    :mm...m(mm...m) nnnnnnnnnn(nn...n)
            Column ID     :oooooo(oo...o)
            Predicate     :pp...p
            Threshold     :qq...q
         [1] rr...r ..............................................3
           Index Name :tt...t uuuuuuuuuu(uu...u) (vv...v) ww...w
                          SearchCnd :xx...x[yy...y],...
                          KeyCnd    :zz...z
           RowCnd     :{GG...G}
           IfThenCnd  :{HH...H}
         [2] rr...r ..............................................4
           RowCnd     :{GG...G}
           IfThenCnd  :{HH...H}
```

Explanation

The `RDAREA` row is not displayed for retrieval processing information for a foreign table.

1 indicates the table retrieval method used when the first candidate is selected for the join method.

2 indicates the table retrieval method used when the second candidate is selected for the join method (`SELECT-APSL`).

3 indicates the first candidate for the table retrieval method when the second candidate is selected for the join method.

4 indicates the second candidate for the table retrieval method when the second candidate is selected for the join method.

aa...a (*aa...a*)

Displays the name of the table (correlation name) being retrieved.

The correlation name is not displayed if it is not used. If retrieval processing is executed more than once, information about each retrieval processing is separated by this line.

bbbbbbbbbb (*bb...b*)

Displays in hexadecimal and in decimal (the latter in parentheses) the ID of the table being retrieved.

c

Displays whether or not optimization information is to be collected by the optimizing information collection utility (`pdgetcst`):

`Y`: Optimization information collected

`N`: Optimization information not collected

dd...d

Displays the number of table rows used by HiRDB to determine the access path.

The default value depends on the SQL optimization mode:

- Cost-based optimization mode 1 or cost-based optimization mode 2 with a HiRDB/Single Server

  10 million lines

- Cost-based optimization mode 1 with a HiRDB/Parallel Server

  10 million lines × number of back-end servers containing the table storage RDAREAs

By executing the optimizing information collection utility (`pdgetcst`), you can change the number of table rows to be used by HiRDB to determine the access path.

ee...e

Displays the number of columns into which a table is divided.

ff...f

Displays the type of table partitioning:

`NON DIVISION`: not partitioned

`KEY RANGE`: key range partitioning (with storage conditions specified)

`PARTITION`: key range partitioning (with boundary values specified)

`MULTIDIM PARTITION`: matrix partitioning (with boundary values specified)

`MULTIDIM`: Matrix partitioning including hash partitioning

`FLEXIBLE HASH`: flexible hash partitioning without using the rebalancing facility

`FIX HASH`: fix hash partitioning without using the rebalancing facility

`RB FLEXIBLE HASH`: flexible hash partitioning using the rebalancing facility

`RB FIX HASH`: fix hash partitioning using the rebalancing facility

gg...g

Displays the hash function during hash partitioning. This information is not displayed when hash partitioning is not performed. This information is not displayed for matrix partitioning that includes hash partitioning.

`(`*hh...h*`RD/`*ii...i*`BES)`

hh...h displays the number of table partitions (number of RDAREAs) and *ii...i* displays the number of back-end servers containing these table storage RDAREAs.

For a non-partitioned table, *hh...h* displays a value of 1. `/`*ii...i*`BES` is not displayed for a HiRDB/Single Server.

jjjj `(`*jj...j*`)`

Displays the ID of the table storage RDAREA in hexadecimal and in decimal (the latter in parentheses).

If the table is partitioned, the utility displays as many RDAREA IDs as there are table partitions. To obtain the name of the RDAREA associated with a given RDAREA ID, execute the `pddbls` command.

kk...k

`ALL`

This is displayed when all RDAREAs defining a table are to be retrieved.

`RESTRICTED`

This is displayed when, among the RDAREAs defining the table, only those RDAREAs that can potentially contain the target data based on the specified search conditions are to be retrieved.

ll...l

Displays the rebalancing status for a table using the rebalancing facility.

This line is not displayed for a table that does not use the rebalancing facility. For details about the rebalancing facility, see the *HiRDB Version 8 System Operation Guide*.

`NORMAL`

This is the normal status or the status when `ALTER TABLE` was used to add an RDAREA but the rebalancing utility (`pdrbal`) has not been executed yet.

`ON REBALANCE`

This is the status when the table is being rebalanced (from the start to the end of the rebalance operation).

mm...m (*mm...m*)

Displays the table name (correlation name) for the column used in the predicate that is subject to calculation of the SQL run-time hit rate when selecting an SQL object.

nnnnnnnnnn (*nn...n*)

Displays in hexadecimal and in decimal (the latter in parentheses) the table ID for the column used in the predicate that is subject to calculation of the SQL run-time hit rate when selecting an SQL object.

oooooo (*oo...o*)

Displays in hexadecimal and in decimal (the latter in parentheses) the column ID used in the predicate that is subject to calculation of the SQL run-time hit rate when selecting an SQL object.

pp...p

Displays the predicate subject to calculation of the SQL run-time hit rate when selecting an SQL object.

qq...q

Displays the reference hit rate for selecting an SQL object.

If the hit rate obtained from calculation is less than this value, the first candidate is used for the SQL object; if it is equal to or greater than this value, the second candidate is used for the SQL object.

rr...r

Displays the retrieval method to be used.

For details about the types of retrieval methods, see *17.5.7 Types of retrieval methods*.

ss...s

Displays the join method corresponding to the retrieval processing.

For details about the types of join methods, see *17.5.6 Types of join methods*.

tt...t

Displays the name of the index used for retrieval.

If no index is used for retrieval, the utility does not display this line.

uuuuuuuuuu (*uu...u*)

Displays the index ID used for retrieval in hexadecimal and in decimal (the latter in parentheses).

vv...v

Displays a combination of index attributes:

Numeric character: Number of columns composing the index

`G`: Plug-in index

`U`: Unique key index

`C`: Cluster key index

`D`: Partition key index or plug-in index

`E`: Index including an exceptional key value

`M`: Index containing repetition columns

`P`: Primary key index

ww...w

The utility displays the index component column information by separating each column by a comma and enclosing the entire information in parentheses.

The index component column information consists of the search direction and the name of the index component column.

As the search direction, + means that the component column is searched in ascending order while - means that it is searched in descending order.

Example

Searching component column 1 (`C1`) in ascending order and component column 2 (`C2`) in descending order:

```
(+C1,-C2)
```

xx...x[*yy...y*], . . .

Displays the type and narrowed range of a search condition. For details about the search conditions, see *17.5.12 Search conditions*.

If there is no search condition for a retrieval using an index, the utility displays `NONE(FULL SCAN)` as the type of search condition.

The number of narrowed ranges may become 0 or 1 depending on the search condition. If there is no search condition for the first component column, but there are search conditions for the subsequent columns, the utility displays `(FULL SCAN)` following the narrowed range of search conditions.

zz...z

Displays the key condition. For details about key conditions, see *17.5.13 Key conditions*.

If there is no key condition, the utility does not display this line.

AA...A

If a shared table is the search target, the utility displays `SHARED`; if not, the utility does not display this item.

(5) Work table created for a view table retrieval

```
SCAN
 # Table Name :aa...a(aa...a) bbbbbbbbbb(bb...b)  {AA...A}
   Cost       :(cc...cROW)  {T-BB...B}
   Scan Type  :dd...d
   RowCnd     :{CC...C}
   IfThenCnd  :{DD...D}
```

Explanation

aa...a(*aa...a*)

Displays the name of the view table (correlation name).

The correlation name is not displayed if it is not used.

bbbbbbbbbb(*bb...b*)

Displays in hexadecimal and in decimal (the latter in parentheses) the ID of the view table.

cc...c

Displays the number of table rows to be used by HiRDB to determine the access path.

dd...d

Displays `LIST SCAN`.

For details about `LIST SCAN`, see *17.5.7 Types of retrieval methods*.

(6) Work table created for a WITH clause

```
SCAN
 # Table Name :aa...a(aa...a)  {AA...A}
   Cost       :(bb...bROW)  {T-BB...B}
   Scan Type  :cc...c
   RowCnd     :{CC...C}
   IfThenCnd  :{DD...D}
```

Explanation

aa...a (*aa...a*)

Displays the name of the `WITH` clause query (correlation name).

The correlation name is not displayed if it is not used.

bb...b

Displays the number of table rows to be used by HiRDB to determine the access path.

cc...c

Displays `LIST SCAN`.

For details about `LIST SCAN`, see *17.5.7 Types of retrieval methods*.

(7) Work table created for the derived table specified in the FROM clause

```
SCAN
 # Table Name :aa...a(aa...a) {AA...A}
   Cost       :(bb...bROW)  {T-BB...B}
   Scan Type  :cc...c
   RowCnd     :{CC...C}
   IfThenCnd  :{DD...D}
```

Explanation

aa...a (*aa...a*)

Displays `(NO NAME)` or `(NO NAME)(`*correlation-name*`)`.

bb...b

Displays the number of table rows to be used by HiRDB to determine the access path.

cc...c

Displays `LIST SCAN`.

For details about `LIST SCAN`, see *17.5.7 Types of retrieval methods*.

(8) Work table created internally by HiRDB

```
SCAN
 # Table Name :aa...a   {AA...A}
   Cost       :(bb...bROW)  {T-BB...B}
   Scan Type  :cc...c
   RowCnd     :{CC...C}
   IfThenCnd  :{DD...D}
```

Explanation

aa...a

Displays the name of the work table created internally by HiRDB.

`(DUMMY` *work-table-number*`)` is the name of a temporary work table created internally by HiRDB. The work table number is a three-digit integer.

bb...b

Displays the number of table rows to be used by HiRDB to determine the access path.

cc...c

Displays `LIST SCAN`.

For details about `LIST SCAN`, see *17.5.7 Types of retrieval methods*.

(9) When retrieving the result of a query to a foreign server

```
SCAN
 # Table Name :aa...a
   Cost       :(bb...bROW){AA...A}
   Scan Type  :cc...c
   RowCnd     :{BB...B}
   IfThenCnd  :{DD...D}
```

Explanation

aa...a

Displays the table identifier `FOREIGNSQL` *table-number* that was created internally in order to receive the results of a retrieval conducted on the foreign server from the local HiRDB.

table-number is a 3-digit integer assigned sequentially beginning at 1.

bb...b

Displays the predicted number of rows containing the retrieval result from the foreign server. This value is used to determine the access path at the local HiRDB.

For details about the predicted number of rows containing the retrieval result from the foreign server, see the *HiRDB Version 8 UAP Development Guide*.

cc...c

Displays the method for retrieving the result of the query to the foreign server.

For details about how to retrieve the results of queries to foreign servers, see *17.5.7 Types of retrieval methods*.

(10) SELECT-APSL used with a HiRDB/Single Server

```
--- SELECT-APSL ---
  Table Name     :aa...a(aa...a) bbbbbbbbbb(bb...b)
  Column ID      :cccccc(cc...c)
  Predicate      :dd...d
  Threshold      :ee...e
[1] .....................................................1
Section No     :ff...f
   :
[2] .....................................................2
Section No     :ff...f
   :
```

Explanation

1 and *2* indicate the first candidate and second candidate, respectively, for the access path. For details about the types of access paths, see *17.5.9 Types of access paths*.

For a HiRDB/Single Server, the single server creates two SQL objects for the entire SQL statement without executing `SELECT-APSL` for each join and retrieval processing.

aa...a(aa...a)

Displays the table name (correlation name) for the column used in the predicate that is subject to calculation of the SQL run-time hit rate when selecting an SQL object.

bbbbbbbbbb(bb...b)

Displays in hexadecimal and in decimal (the latter in parentheses) the table ID for the column used in the predicate that is subject to calculation of the SQL run-time hit rate when selecting an SQL object.

cccccc(cc...c)

Displays in hexadecimal and in decimal (the latter in parentheses) the column ID for the column used in the predicate that is subject to calculation of the SQL run-time hit rate when selecting an SQL object.

dd...d

Displays the predicate subject to calculation of the SQL run-time hit rate when selecting an SQL object.

ee...e

Displays the reference hit rate for selecting an SQL object.

If the hit rate obtained from calculation is less than this value, the first candidate is used for the SQL object; if it is equal to or greater than this value, the second candidate is used for the SQL object.

ff...f

Displays the section number (assigned to each SQL statement).

For a dynamic SQL statement, this value is 1.

17.5.6 Types of join methods

In the editing result, `L` indicates an outer table and `R` indicates an inner table. For details about the join methods, see the *HiRDB Version 8 UAP Development Guide*.

(1) Merge join

The merge join method is effective when the outer table cannot be narrowed down significantly.

`SORT MERGE JOIN`

This method retrieves rows from the outer and inner tables, creates a work table for each of them, sorts the rows in the work tables, then joins the rows satisfying the join conditions.

`KEY SCAN MERGE JOIN`

This method retrieves rows from the outer and inner tables by `KEY SCAN` and joins the rows satisfying the join conditions.

`LIST SCAN MERGE JOIN`

This method creates work tables from the outer and inner tables, retrieves rows in ascending order of the joined columns without sorting, then joins the rows satisfying the join conditions.

`L-KEY R-LIST MERGE JOIN`

This method retrieves rows from the outer table by `KEY SCAN`, creates a work table for the inner table and retrieves rows without sorting, then joins the rows satisfying the join conditions.

`L-KEY R-SORT MERGE JOIN`

The method retrieves rows from the outer table by `KEY SCAN`, creates a work table for the inner table and retrieves rows after sorting, then joins the rows satisfying the join conditions.

`L-LIST R-KEY MERGE JOIN`

This method creates a work table for the outer table and retrieves rows without sorting, retrieves rows from the inner table by `KEY SCAN`, then joins the rows satisfying the join conditions.

`L-LIST R-SORT MERGE JOIN`

This method creates a work table for the outer table and retrieves rows without sorting, creates a work table for the inner table and retrieves rows after sorting, then joins the rows satisfying the join conditions.

`L-SORT R-KEY MERGE JOIN`

This method creates a work table for the outer table and retrieves rows after sorting, retrieves rows from the inner table by `KEY SCAN`, then joins the rows satisfying the join conditions.

`L-SORT R-LIST MERGE JOIN`

This method creates a work table for the outer table and retrieves rows after sorting, creates a work table for the inner table and retrieves rows without sorting, then joins the rows satisfying the join conditions.

(2) Nested loops join

The nested loops join method is effective when an index is defined for the inner table and the outer table can be narrowed down.

`NESTED LOOPS JOIN`

This join method involves nested loop processing; that is, it retrieves one row at a time from the outer table, matches it with each row in the inner table, then retrieves the row if it satisfies the join conditions.

`R-LIST NESTED LOOPS JOIN`

This method first retrieves rows from the inner table and creates a work table. Next, it retrieves one row at a time from the outer table, matches it with each row of the work table that was created from the inner table, then retrieves the row if it satisfies the join conditions.

(3) Hash join

`HASH JOIN{FOR EACH}`

This method first creates a hash table by hashing with the value of joined columns in the inner table. Next, it retrieves one row at a time from the outer table and hashes it with the value of joined columns in the outer table, then matches the row with the hash table created from the inner table to join the row.

(4) SELECT-APSL

`SELECT-APSL` is a method for dynamically determining the join method during SQL execution.

`SELECT-APSL` (applicable only to HiRDB/Parallel Server)

If the specified conditions contain a `?` parameter, the optimum join method may depend on the value of the `?` parameter. The optimum join method cannot be selected during SQL optimization processing because the value of the `?` parameter cannot be determined. This method selects a join method by calculating the hit rate during SQL execution.

(5) Distributed nest-loop-join

Distributed nest-loop-join is a method for using nested loop join when a foreign table is different from the inner table.

`DISTRIBUTED NESTED LOOPS JOIN`

This join method uses nested loop processing, which means that it retrieves rows from the outer table, executes an SQL statement to pass a row values in the outer table as a variable to the foreign server that contains the foreign table to be used as the inner table for each retrieved row, and then compares the rows to retrieve only those that satisfy the join conditions.

(6) Direct product

`CROSS JOIN`

This method joins all rows in the outer table and inner table. After joining the tables, it checks any condition involving two or more tables.

17.5.7 Types of retrieval methods

For details about the retrieval methods, see the *HiRDB Version 8 UAP Development Guide*.

(1) Retrieval without using an index

`TABLE SCAN`

This method retrieves data pages in a table without using an index.

(2) Retrieval using one index

`INDEX SCAN`

This method narrows down the table by retrieving the index pages of a single-column index, then retrieves the data pages of the table.

`KEY SCAN`

This method retrieves only the index pages of a single-column index. It does not retrieve data pages.

`MULTI COLUMNS INDEX SCAN`

This method narrows down the table by retrieving the index pages of a multicolumn index, then retrieves the data pages of the table.

`MULTI COLUMNS KEY SCAN`

This method retrieves only the index pages of a multicolumn index. It does not retrieve data pages.

`PLUGIN INDEX SCAN`

This method retrieves table data pages after narrowing the search by using a plug-in index.

`PLUGIN KEY SCAN`

This method retrieves index pages by using a plug-in index only. It does not retrieve data pages.

(3) SELECT-APSL

`SELECT-APSL` (applicable only to HiRDB/Parallel Server)

If the specified conditions contain a `?` parameter, the optimum join method may depend on the value of the `?` parameter. The optimum join method cannot be selected during preprocessing because the value of the `?` parameter cannot be determined. This method selects a join method by calculating the hit rate during SQL execution.

(4) Retrieval using a multicolumn index

`AND PLURAL INDEXES SCAN`

This method retrieves rows using each applicable index according to the search conditions concatenated by `AND` and `OR` operators, and stores the row identifiers (`ROWID`) in each work table. The method combines all work tables into one by obtaining the product set for the `AND` operators, the union for the `OR` operators, and the difference set for the `ANDNOT` operators (specifiable only in the `ASSIGN LIST` statement). Then it retrieves rows on the basis of the row identifiers in the work table.

When a work table of row identifiers is created from a given set of conditions, the utility may create the work table using `TABLE SCAN`, even when there is no index for condition columns.

`OR PLURAL INDEXES SCAN`

This method retrieves rows using each index according to the search conditions concatenated by `OR` operators, and stores row identifiers (`ROWID`) in a single work table. The method eliminates all duplicate rows from the work table, then retrieves rows on the row identifier.

When a work table for row position information is created from a given set of conditions, the utility may create the work table using `TABLE SCAN`, even when there is no index for condition columns.

(5) Retrieval of work tables

`LIST SCAN`

This method retrieves the work tables there are created internally.

(6) Retrieval using row identifier

`ROWID FETCH`

This method searches a table by using the row identifier (`ROWID`) as a key. The system does not execute this search if there is no need to fetch rows.

(7) Retrieval of the result of queries to foreign servers

`FOREIGN SERVER SCAN`

This method issues SQL statements to foreign servers to receive the results of a query.

`FOREIGN SERVER LIMIT SCAN`

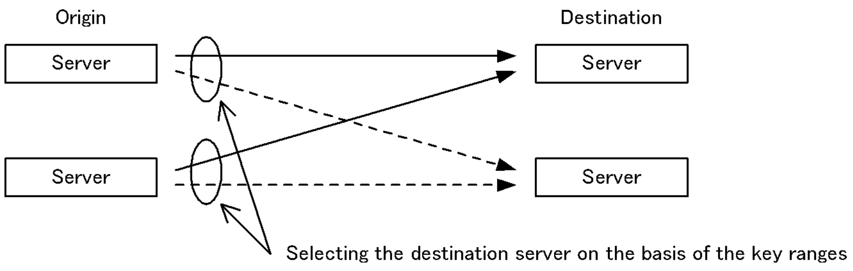
This may be displayed when the function for retrieving the first *n* rows of the retrieval result is used. This method issues an SQL statement containing the `ORDER BY` clause to a foreign server and receives the first *n* rows of the query result.

17.5.8 Types of transfer methods

(1) Key range transfer

`KEY RANGE`

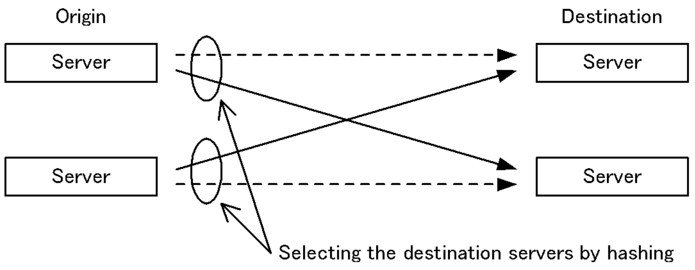
This method selects the destination servers on the basis of key ranges.


Origin
Destination
Server
Server
Server
Server
Selecting the destination server on the basis of the key ranges

(2) Hash transfer

`HASH`

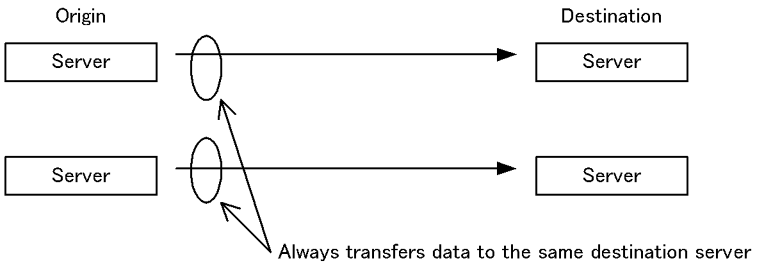
This method selects the destination servers by hashing.



(3) One-to-one transfer

`1 TO 1`

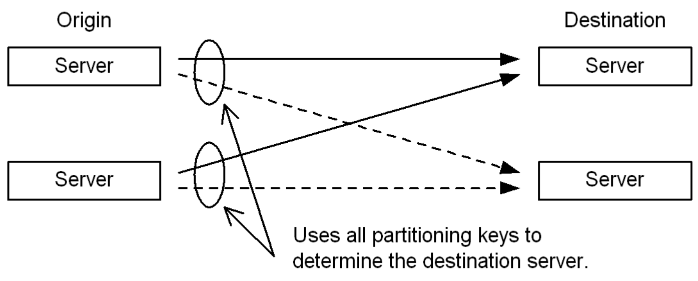
This method selects the destination server that has a one-to-one relationship with the server at the origin.



(4) Matrix partitioning transfer

`MULTIDIM`

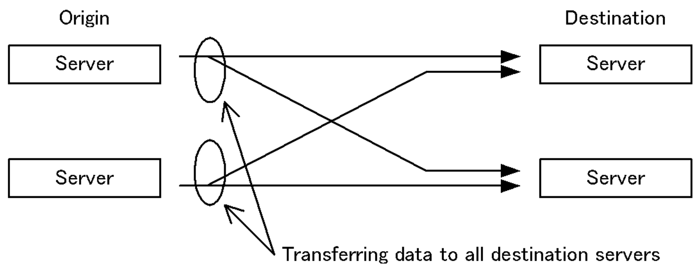
This transfer method determines the target server using all partitioning keys of a matrix-partitioned table, including hash partitioning.


Origin
Destination
Server
Server
Server
Server
Uses all partitioning keys to determine the destination server.

(5) Broadcast transfer

`BROADCAST`

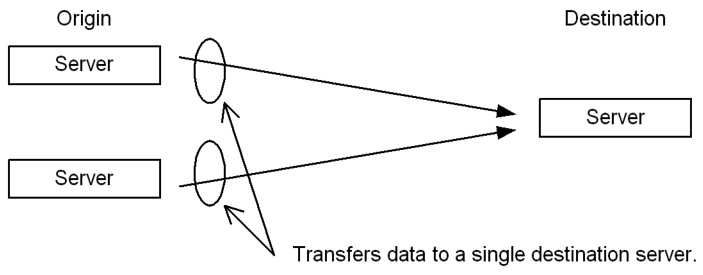
This method copies the data to be transferred and sends it to all servers at the destination.



(6) Unicast transfer

`UNICAST`

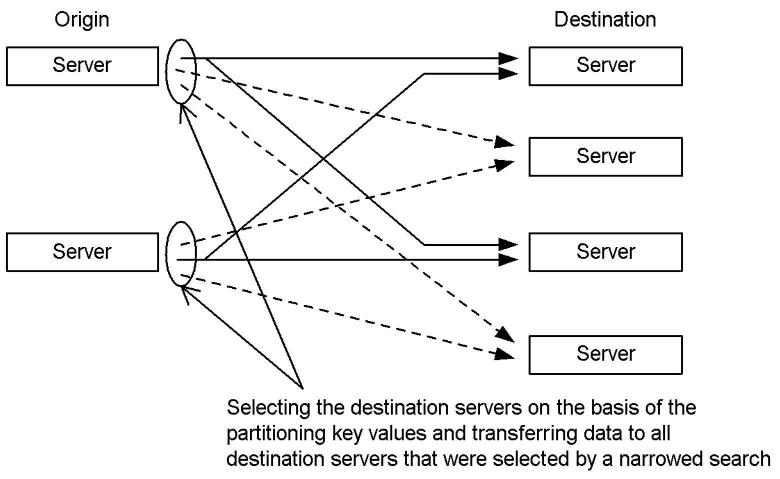
This method transfers data from the origin to a single destination server. The utility displays `BROADCAST` except for the `Join Type` section in the join processing information.



(7) Partial key range broadcast transfer

`KEY RANGE PARTIAL BROADCAST`

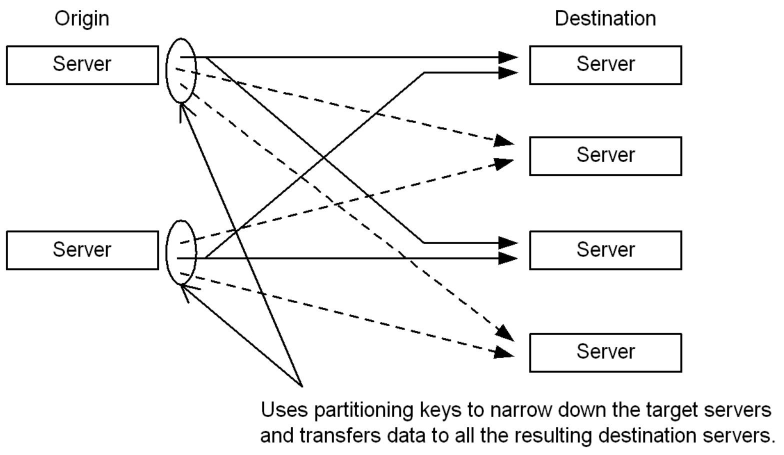
This method uses partial key ranges of a matrix-partitioned (boundary values) table to narrow down the destination servers, copies the origin data, and then sends it to all the resulting destination servers.


Origin
Destination
Server
Server
Server
Server
Server
Server
Selecting the destination servers on the basis of the partitioning key values and transferring data to all destination servers that were selected by a narrowed search

(8) Partial broadcast transfer

`PARTIAL BROADCAST`

This method uses part of the partitioning keys of a matrix-partitioned table including hash partitioning to narrow down the destination servers, copies the origin data, and then sends it to all the resulting destination servers.


Origin
Destination
Server
Server
Server
Server
Server
Server
Uses partitioning keys to narrow down the target servers and transfers data to all the resulting destination servers.

17.5.9 Types of access paths

`SELECT-APSL` (applicable only to HiRDB/Single Server)

If the specified conditions contain a `?` parameter, the optimum access path may depend on the value of the `?` parameter. The optimum access path cannot be selected during SQL optimization processing because the value of the `?` parameter cannot be determined. This method selects an access path by calculating the hit rate during SQL execution.

17.5.10 Types of grouping processing methods

- Implicit grouping

  This method specifies a set function in either a selection expression or a `HAVING` clause without specifying a `GROUP BY` clause in the query expression.

- Explicit grouping

  This method specifies a `GROUP BY` clause in the query expression.

(1) HiRDB/Single Server

(a) Implicit grouping

`IMPLICIT MIN-MAX INDEX`

This method retrieves only the minimum and the maximum values of the index when determining the set functions `MIN` and `MAX`.

`IMPLICIT SET FUNCTION SCAN{+}`

This method obtains the result of the set function at the same time as retrieving a row.

`IMPLICIT NORMAL`

This method obtains the result of the set function after retrieving a row.

`IMPLICIT LIST SORT{SET FUNCTION SCAN}`

If `DISTINCT` is included in the set function, this method creates a work table and sorts it to eliminate duplicate values before obtaining the result of the set function.

`IMPLICIT SURROGATE(COUNT)`

This method obtains the result of the set function (number of rows) by using the surrogate function. The surrogate function achieves special processing; that is, it evaluates search conditions by calling a plug-in index and simultaneously determines the result of the set function internally in a plug-in-supplied function to rapidly return the results of the set function.

`IMPLICIT SORT CANCEL BY INDEX{SET FUNCTION SCAN}`

If the column specified in the `DISTINCT` set function can be sorted by searching the index without having to perform sort processing for the purpose of eliminating duplicates, this method omits the sort processing and obtains the result of the set function.

(b) Explicit grouping

`SORT CANCEL BY INDEX{SET SCAN}`

This method skips the sorting for grouping in situations in which the columns to be grouped can be sorted by retrieving their index.

`SORT CANCEL BY JOIN`

This method skips the sorting for grouping in situations in which the sorting for grouping purposes is cancelled by merge-join sorting.

`HASH`

This method uses the rapid grouping facility. For details about the rapid grouping facility, see the *HiRDB Version 8 UAP Development Guide*.

`LIST SORT`

This method obtains the result of grouping by creating a work table and sorting it.

`PRE-SORT JOIN`

In situations in which all the columns to be grouped are included in an outer table before it is joined or in an outer-joined table, this method sorts the pre-joined outer table or the result of the outer join, then performs grouping by joining the tables while maintaining their sort order, thereby determining the result of grouping.

(2) HiRDB/Parallel Server

(a) Implicit grouping

`IMPLICIT MIN-MAX INDEX`

This method retrieves only the minimum and the maximum values of the index on each back-end server when determining the set functions `MIN` and `MAX`.

`IMPLICIT SET FUNCTION SCAN{+}`

This method obtains the result of the set function on each back-end server at the same time as retrieving a row.

`IMPLICIT NORMAL`

This method obtains the result of the set function after retrieving a row on each back-end server.

`IMPLICIT FLOATABLE`

This method obtains the results of all set functions by collecting the results on a floatable server.

`IMPLICIT FLOATABLE SORT{SET FUNCTION SCAN}`

If `DISTINCT` is included in the set function, this method collects the results on a floatable server, creates a work table, and sorts them to eliminate duplicate values before obtaining the result of the set function.

`IMPLICIT SURROGATE(COUNT)`

This method determines the result of the set function (number of rows) for each back-end server by using the surrogate function. The surrogate function achieves special processing; that is, it evaluates search conditions by calling a plug-in index and simultaneously determines the result of the set function internally in a plug-in-supplied function to rapidly return the results of the set function.

`IMPLICIT SORT CANCEL BY INDEX{SET FUNCTION SCAN}`

If the column specified in the `DISTINCT` set function can be sorted by searching the index without having to perform sort processing for the purpose of eliminating duplicates, this method omits the sort processing and obtains the result of the set function.

`IMPLICIT LIST SORT{SET FUNCTION SCAN}`

If the set functions include `DISTINCT`, this method creates a work table at each back-end server, performs sorting to eliminate duplicates, and then obtains the result of the set function.

(b) Explicit grouping

`SORT CANCEL BY INDEX{SET SCAN}`

This method skips the sorting for grouping in situations in which the columns to be grouped can be sorted by retrieving their index.

`SORT CANCEL BY JOIN`

This method skips the sorting for grouping in situations in which the sorting for grouping purposes is cancelled by merge-join sorting.

`HASH`

This method obtains the result of grouping by using the rapid grouping facility. For details about the rapid grouping facility, see the *HiRDB Version 8 UAP Development Guide*.

`LIST SORT`

This method obtains the result of grouping by creating a work table on each back-end server and sorting the work table.

`PRE-SORT JOIN`

In situations in which all the columns to be grouped are included in an outer table before it is joined or in an outer-joined table, this method sorts the table, then performs grouping by joining the tables while maintaining their sort order, thereby determining the result of grouping.

`FLOATABLE SORT`

This method obtains the result of grouping by transferring data to multiple floating servers, creating a work table, and sorting the data in it.

17.5.11 Types of plug-in-provided functions

The plug-in-provided functions that use an index-type plug-in for retrieval are classified into three types according to plug-in index utilization conditions described as follows.

(1) SCAN TYPE plug-in-provided function

This type of plug-in-provided function uses a logical predicate to specify search conditions.

Normally a plug-in index is used for retrieval. However, a `SCAN TYPE` plug-in-provided function can work even when no plug-in index is defined, or when a plug-in index is not available during retrieval.

(2) INDEX SCAN TYPE plug-in-provided function

This type of plug-in-provided function uses a logical predicate (`IS TRUE` only) to specify search conditions. It always uses a plug-in index for retrieval. This is called an index-type plug-in-dedicated function.

An `INDEX SCAN TYPE` plug-in-provided function results in an error if a plug-in index is undefined or unavailable.

(3) FULL SCAN TYPE plug-in-provided function

This type of plug-in-provided function uses a logical predicate to specify search conditions.

If a plug-in index is defined, the index can be used for retrieval, but the entire range of index is searched. Therefore, if the condition is provided only by the `FULL SCAN TYPE` plug-in-provided function, the plug-in index is used for retrieval. If there is another condition, the function checks that condition to obtain the result without using the plug-in index.

17.5.12 Search conditions

(1) Overview of search conditions

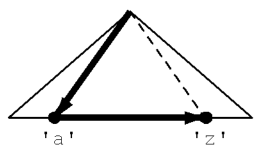
`SearchCnd:`*search-condition-type*`[`*narrowed-range*`],...`

A search condition determines the range of index to be searched.

(a) Search condition specified

Example

```
where C1 between 'a' and 'z'
```



```
SearchCnd : RANGE(CS-CE)['a','z']
```

Note

The index component column to be used is `C1`.

(b) Search condition deleted during execution

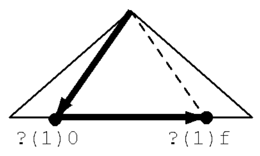
If you specify an embedded variable, `?` parameter, SQL variable, `LIKE` predicate containing an SQL parameter, or a `SIMILAR` predicate in a pattern character string, and the pattern character string given during execution results in right truncation, such as `abc%`, the utility can narrow the index search range. However, if the pattern character string results in left truncation such as `%abc`, the utility cannot narrow the index search range.

If the index search range cannot be narrowed due to the value of pattern character string provided during execution, the utility displays in diamond brackets (< >) the start and end values used to narrow the index range.

Example

```
where C1 LIKE ?
```

Where the data type of `C1` is `CHAR(5)`.



```
SearchCnd : RANGE(CS-CE)[<?(1)0>,<?(1)f>]
```

Note

The index component column to be used is `C1`.

Explanation:

- If `?` is `abc%`, the utility searches the range from `X'6162630000'` to `X'616263ffff'`, where the character code `a` is `X'61'`, `b` is `X'62'`, and `c` is `X'63'`.

- If `?` is `%abc`, the utility searches the entire range of the index.
- `?(1)` indicates the `?` number. For details, see (3) *Values used in a narrowed range*.

(2) Types of search conditions

Table 17-1 lists the search conditions by type.

Table 17-1: Search conditions by type

Type of search condition	Characteristics	Condition for creating the type of search condition	
		Single-column index	Multicolumn index
`IS NULL`	Effective when there are only a few null key values.	There is a `NULL` predicate (`IS NULL`) in all index component columns.	
`IS NOT NULL`	Effective when there are only a few non-null key values.	There is a `NULL` predicate (`IS NULL`) in the index component columns.	Not created
`IS TRUE`	Depends on the plug-in-provided function.	There is checking using a Boolean predicate (`IS TRUE`) for the execution result of a plug-in-provided function.	Not created
`AT`	Effective when there are only a few duplicates in the narrowing value.	There is a comparison predicate (=) in one or more index component columns and a `NULL` predicate (`IS NULL`) in all other index component columns.	
`RANGE`	Effective when there are only a few narrowed ranges.	The index component column contains a comparison predicate (>, >=, <, <=), `BETWEEN` predicate, `LIKE` or `SIMILAR` predicate that results in right truncation. For a multicolumn index, there is a comparison predicate (=) or a `NULL` predicate (`IS NULL`) in some of the index component columns. This does not apply to `RANGES`.	

Type of search condition	Characteristics	Condition for creating the type of search condition	
		Single-column index	Multicolumn index
ATS	Effective when there are only a few duplicates in the narrowing value and only a few narrowing values.	There is an IN (*row-value-constructor*[3]), IN (*table-subquery*), =ANY (*table-subquery*), or =SOME (*table-subquery*) in one or more index component columns,[1] and either a comparison predicate (=) or an IS NULL in all the other index component columns. If the product of the narrowing values specified in IN (*row-value-constructor*[3]), IN (*table-subquery*), =ANY (*table-subquery*), or =SOME (*table-subquery*) exceeds 255, RANGES is assumed.[2]	
RANGES	Effective when there are only a few items in the narrowed ranges and only a few narrowed ranges.	Note created	The index component column containing an IN (*row-value-constructor*[3]), IN (*table-subquery*), =ANY (*table-subquery*), or =SOME (*table-subquery*) is the first index component column;[1] or, if it is not the first index component column, all the index component columns preceding this index component column contain either comparison predicate (=) or IS NULL. This does not apply to ATS.

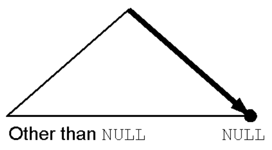
[1] If the SQL optimization mode being used is the optimizing mode based on cost 2, IN(*table-subquery*), =ANY(*table-subquery*), or =SOME(*table-subquery*) uses only one column for narrowing down the search range. If a multicolumn index is used and the SQL optimization mode in use is the optimizing mode based on cost 1, IN(*table-subquery*), =ANY(*table-subquery*), or =SOME(*table-subquery*) is not used for narrowing the search range.

[2] The number of narrowing values for IN(*row-value-constructor*) is the number of row value constructors. It is 44 for IN(*table-subquery*), =ANY(*table-subquery*), and =SOME(*table-subquery*).

[3] This is applicable if the row value constructor elements of all row value constructors are value specification, scalar subquery, or row subquery.

(a) IS NULL

IS NULL searches a range of index in which the value is null.



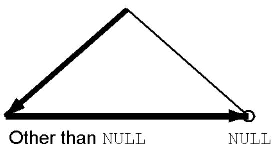
Note

The index component column to be used is `C1`.

Example
```
where C1 IS NULL
  →   SearchCnd: IS NULL
```

(b) IS NOT NULL

`IS NOT NULL` searches a range of index in which the value is not null.



Note

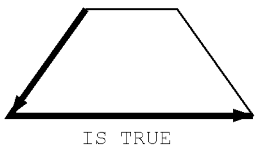
The index component column to be used is `C1`.

Example
```
where C1 IS NOT NULL
  →   SearchCnd: IS NOT NULL
```

(c) IS TRUE

`IS TRUE` is displayed for a retrieval using a plug-in index.

FULL SCAN TYPE plug-in-provided function



The structure of a plug-in index and the narrowing method depend on the plug-in.

Note

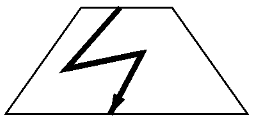
The index component column to be used is `feature`.

Example

```
where SearchImageData (feature,? as blob(1M),
                       ? as varchar(1024))IS TRUE
```

→ `SearchCnd: IS TRUE`

Other than FULL SCAN TYPE plug-in-provided function



The structure of a plug-in index and the narrowing method depend on the plug-in.

Note

The index component column to be used is `SENTENCES`.

Example

```
where Contains (SENTENCES,'DOCUMENT_DATA[EFFECTS
                {"FOOD_POISONING"}]')IS TRUE
```

→ `SearchCnd: IS TRUE`

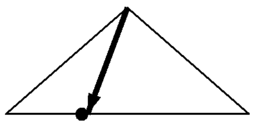
(d) AT

Search condition type

`AT[`*narrowing-value*`]:`

Use `AT` to search the index for a specific value.

The search condition can be `C1='a'`, `C1=(Select C1 from T2)`, etc.



Note

The index component column to be used is `C1`.

Example

```
where C1='a'
    →  SearchCnd: AT ['a']
```

Narrowing value

`AT` for a single-column index

`AT[`*value*`]`:

Displays the value for a single-column index.

Example

```
where C1='a'
    →  SearchCnd: AT ['a']
```

Note: The index component columns to be used is `C1`.

`AT` for a multicolumn index

`AT[(`*value*`,...,`*value*`)]`:

For a multicolumn index, the utility displays the values sequentially from the first index component column by separating the values with the comma (`,`) and enclosing the entire set of values in parentheses.

Example 1

```
where C1='a' and C2='A'
    →  SearchCnd: AT [('a','A')]
```

Note: The index component columns to be used are `C1` and `C2`.

Example 2

```
where (C1,C2) in (('a','A'))
        →  SearchCnd: AT [('a','A')]
```

Note: The index component columns to be used are `C1` and `C2`.

Example 3

```
where (C1,C2) = ((select C1 from T2),(select C2 from T3))
        →  SearchCnd: AT [((SUBQ(2)),(SUBQ(3)))]
```

Note: The index component columns to be used are `C1` and `C2`.

Example 4

```
where array(C1,C2) [any] ((C1,C2)=('a','A'))
        →  SearchCnd: AT [('a','A')]
```

Note: The index component columns to be used are `C1` and `C2`.

(e) RANGE

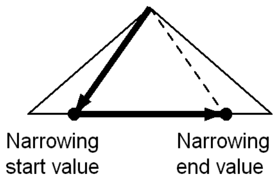
Search condition type

Use `RANGE` to search an index in a narrowed range from the start value to the end value. `RANGE` is classified into four types depending on whether it includes the start value for the narrowed range and whether it includes the end value for the narrowed range. This information is displayed in the portion enclosed in double quotation marks (`""`) in `RANGE(...)`.

1. Both start and end values of narrowed range included

   `RANGE(CS_CE)[`*narrowing-start-value*`,`*narrowing-end-value*`]`:

   ```
   The search condition can be C1>='a' and C1<='z', C1 between
   'a' and 'z', C1 like 'a%', etc.
   ```



   Example 1

   ```
   where C1 >= 'a' and C1 <= 'z'
           →  SearchCnd: RANGE(CS-CE) ['a','z']
   ```

   Note: The index component column to be used is `C1`.

   Example 2

   ```
   where C1 >= 'a'
           →  SearchCnd: RANGE(CS-CE) ['a',MAX]
   ```

   Note: The index component column to be used is `C1`.

   Example 3

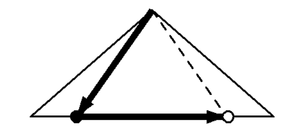
   ```
   where C1 <= 'a'
           →  SearchCnd: RANGE(CS-CE) [MIN,'a']
   ```

   Note: The index component column to be used is `C1`.

2. Narrowing start value included, but narrowing end value not included

   `RANGE(CS_OE)[`*narrowing-start-value*`,`*narrowing-end-value*`]`:

   The search condition can be `C1>='a' and C1<'z'`, etc.



Example 1
```
where C1 >= 'a' and C1 < 'z'
      →  SearchCnd: RANGE(CS-OE) ['a','z']
```

Note: The index component column to be used is `C1`.

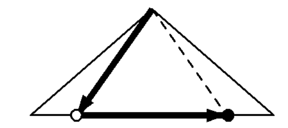
Example 2
```
where C1 < 'a'
      →  SearchCnd: RANGE(CS-OE) [MIN,'a']
```

Note: The index component column to be used is `C1`.

3. Narrowing start value not included, but narrowing end value included

   `RANGE(OS_CE)[`*narrowing-start-value*`,`*narrowing-end-value*`]`:

   The *search* condition can be `C1>='a' and C1<='z'`, etc.



Example 1
```
where C1 > 'a' and C1 <= 'z'
      →  SearchCnd: RANGE(OS-CE) ['a','z']
```

Note: The index component column to be used is `C1`.

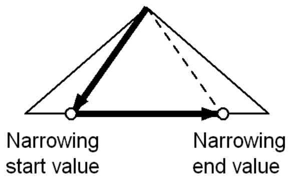
Example 2
```
where C1 > 'a'
      →  SearchCnd: RANGE(OS-CE) ['a',MAX]
```

Note: The index component column to be used is `C1`.

4. Neither narrowing start value nor end value included

`RANGE(OS_OE)[`*narrowing-start-value*`,`*narrowing-end-value*`]`:

The search condition can be `C1>='a' and C1<'z'`, etc.



Example
```
where C1 > 'a' and C1 < 'z'
    →  SearchCnd: RANGE(OS-OE) ['a','z']
```

Note: The index component column to be used is `C1`.

Narrowed range (narrowing-start-value, narrowing-end-value)

`RANGE` for a single-column index

`RANGE (...) [`*value*`,`*value*`]`:

For a single-column index, the utility displays the start and end positions of the narrowed range by separating them by a comma. `...` in `RANGE (...)` depends on the search condition type.

Example
```
where C1 between 'a' and 'z'
    →  SearchCnd: RANGE(CS-CE) ['a','z']
```

Note: The index component column to be used is `C1`.

`RANGE` for a multicolumn index

`RANGE (...) [(`*value*`,...,`*value*`),(`*value*`,...,`*value*`)]`:

For a multicolumn index, the utility displays a set of values sequentially from the first index component column by separating each value by a comma (`,`) and enclosing the entire values in parentheses for the start position and for the end position, which are further separated by a comma. `...` in `RANGE (...)` depends on the search condition type.

Example 1
```
where C1 = 'a' and C2 between 'A' and 'Z'
    →  SearchCnd: RANGE(CS-CE) [('a','A'),('a','Z')]
```

Note: The index component columns to be used are `C1` and `C2`.

Example 2

```
where (C1,C2) between ('A', 'Z') and (select C1, C2 from
T2)
        →   SearchCnd: RANGE(CS-CE)[( 'A',
'Z'),((SUBQ(2)),(SUBQ(2)))]
```

Note: The index component columns to be used are `C1` and `C2`.

Example 3

```
where (C1,C2) > ('a', 'z') and (C1,C2) < ('A', 'Z')
        →   SearchCnd: RANGE(OS-OE)[( 'a', 'z'),( 'A',
'Z')]
```

Note: The index component columns to be used are `C1` and `C2`.

Example 4

```
where array(C1,C2) [any] ((C1,C2)<( 'A', 'Z'))
        →   SearchCnd: RANGE(CS-OE)[(MIN,MIN),( 'A',
'Z')]
```

Note: The index component columns to be used are `C1` and `C2`.

(f) ATS

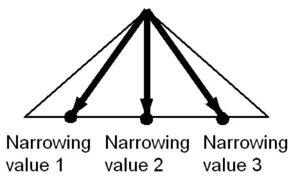
Search condition type

`ATS` with multiple narrowing values

`ATS[`*narrowing-value*`],...,[`*narrowing-value*`]`:

Use `ATS` to search an index more than once using narrowing values.

An example of search condition is `C1 in ('a', 'b', 'c')`.



Example

```
where C1 in ('a','b','c')
     →   SearchCnd: ATS ['a'],['b'],['c']
```

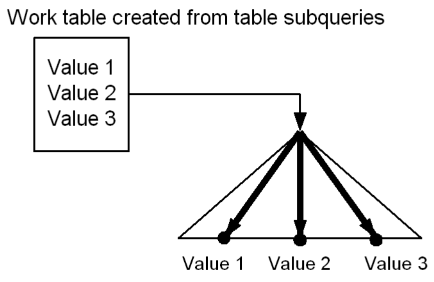
Note: The index component column to be used is `C1`.

ATS with a set of narrowing values

`ATS [`*set-of-narrowing-values*`]:`

Use this type of `ATS` to search an index more than once using a set of narrowing values (work table created from table subqueries).

Examples of search condition are `C1=any(Select C1 from T2)`, `C1=some(Select C1 from T2)`, and `C1 in (Select C1 from T2)`.



Example

```
where C1 =any(select C1 from T2)
```

$\rightarrow$ `SearchCnd:ATS [SUBQ(2)]`

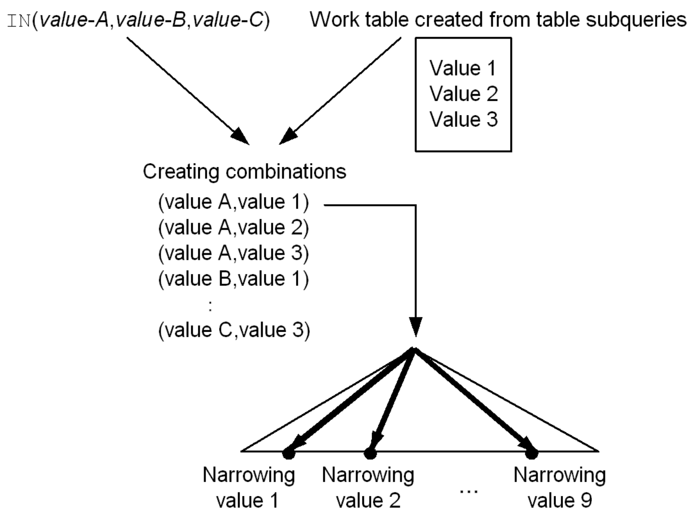
Note: The index component column to be used is `C1`.

ATS with multiple sets of narrowing values

`ATS [`*set-of-narrowing-values*`], ... ,[`*set-of-narrowing-values*`]:`

Use this type of ATS to search an index more than once using the result of combining sets of narrowing values (work table created from table subqueries) and the `IN` predicate.

An example of search condition is `C1 in ('a','b','c') and C2=any(Select C1 from T2)`.



Example

```
where C1 in ('a','b','c') and C2=any(Select C1 from T2)
```

→
```
SearchCnd: ATS [('a',(SUBQ(2)))],
                [('b',(SUBQ(2)))],
                [('c',(SUBQ(2)))]
```

Note: The index component columns to be used are `C1` and `C2`.

- Narrowing value or a set of narrowing values

ATS for a single-column index by multiple narrowing values

`ATS [`*value*`], ... ,[`*value*`]`:

Displays the values separated by the comma.

Example

```
where C1 in ('a','b','c')
```

→
```
SearchCnd:ATS ['a'],['b'],['c']
```

Note: The index component column to be used is `C1`.

ATS for a single-column index by a set of narrowing values

`ATS [`*(SUBQ(query-ID))-or-(SUBQEX(query-expression-ID))*`]`:

Displays the ID of the specified table subquery or query expression.

Example

```
where C1 =any(select C1 from T2)
```

```
        →   SearchCnd: ATS [(SUBQ(2))]
```

Note: The index component column to be used is `C1`.

ATS for a multicolumn index by multiple narrowing values

`ATS [(`*value*`,...,`*value*`)], ... ,[(`*value*`,...,`*value*`)]:`

For a multicolumn index, a single narrowing value consists of the values that are sequentially separated by the comma from the top of the index component columns and then enclosed in its entirety in double quotation marks ("). More than one such narrowing value is displayed by separating them with the comma.

Example 1

```
where C1 in ('a','b') and C2 in ('A','B') and C3=1
        →   SearchCnd:ATS
[('a','A',1)],[('a','B',1)],[('b','A',1)],[('b','B',1)]
```

Note: The index component columns to be used are `C1`, `C2`, and `C3`.

Example 2

```
where (C1,C2) in (('a','b'), ('A','B'))
        →   SearchCnd: ATS [('a','b'),( 'A','B')]
```

Note: The index component columns to be used are `C1` and `C2`.

ATS for a multicolumn index by a set of narrowing values

`ATS [(`*value*`,(`*SUBQ*`(`*query-ID*`))` or `(`*SUBQEX*`(`*query-expression-ID*`)),...,`*value*`,(`*SUBQ*`(`*query-ID*`))` or `(`*SUBQEX*`(`*query-expression-ID*`)))]`

For a multicolumn index, a set of narrowing values consists of values or query IDs of table subqueries that are sequential and separated by the comma from the top of the index component columns and then enclosed in their entirety in double quotation marks (").

Example 1

```
where C1=any(select C1 from T2) and C2='a' and C3='A'
        →   SearchCnd: ATS [((SUBQ(2)),'a','A')]
```

Note: The index component columns to be used are `C1`, `C2`, and `C3`.

Example 2

```
where C1= 'a' and (C2,C3) in (select C2, C3 from T2)
        →   SearchCnd: ATS [('a',(SUBQ(2)),(SUBQ(2)))]
```

Note: The index component columns to be used are `C1`, `C2`, and `C3`.

ATS for a multicolumn index by multiple sets of narrowing values

`ATS [(`*value*`,(`*SUBQ*`(`*query-ID*`))` or `(`*SUBQEX*`(`*query-expression-ID*`)),...,`*value*`,(`*SUBQ*`(`*query-ID*`))` or `(`*SUBQEX*`(`*query-expression-ID*`)))]`

`, ... ,[(`*value*`,(`*SUBQ*`(`*query-ID*`))` or `(`*SUBQEX*`(`*query-expression-ID*`)),...,`*value*`,(`*SUBQ*`(`*query-ID*`))` or `(`*SUBQEX*`(`*query-expression-ID*`)))]`

For a multicolumn index, a set of narrowing values consists of the values or query IDs of table subqueries that are sequential and separated by the comma from the top of the index component columns and then enclosed in their entirety in double quotation marks ("). More than one set of such narrowing values is displayed by separating them with the comma.

Example

```
where C1=any(select C1 from T2) and C2 in ('a','b')
             and C3 in ('A','B')

        →    SearchCnd: ATS
[((SUBQ(2)),'a','A')],[((SUBQ(2)),'a','B')],

[((SUBQ(2)),'b','A')],[((SUBQ(2)),'b','B')]
```

Note: The index component columns to be used are `C1`, `C2`, and `C3`.

(g) RANGES

■ Search condition types

`RANGES` is applicable to multicolumn indexes only. Use it to search a range of index narrowing values, from start value to end value, more than once.

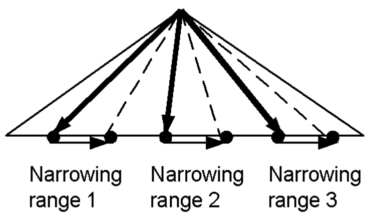
`RANGES` is classified into three types depending on whether it includes the start value for the narrowed range and whether it includes the end value for the narrowed range. This information is displayed in the portion enclosed in double quotation marks ("") in `RANGE(...)`.

RANGES by multiple narrowed ranges

`RANGES(...) [`*narrowing-start-value*`,`*narrowing-end-value*`], ... ,[`*narrowing-start-value*`,`*narrowing-end-value*`]:`

Use this type of `RANGES` to search an index more than once using narrowed ranges.

An example of a search condition is `C1 in ('a','b','c') and C2 between 'A' and 'Z',C1 in ('a','b','c')`.



Example 1

```
where C1 in ('a','b','c') and C2 between 'A' and 'Z'
           →    SearchCnd: RANGES(CS-CE)

[('a','A'),('a','Z')],[('b','A'),('b','Z')],[('c','A'),
('c','Z')]
```

Note: The index component columns to be used are `C1` and `C2`.

Example 2

```
where C1 in ('a','b','c')
           →    SearchCnd: RANGES(CS-CE)

[('a',MIN),('a',MAX)],[('b',MIN),('b',MAX)],
                     [('c',MIN),('c',MAX)]
```

Note: The index component columns to be used are `C1` and `C2`.

Example 3

```
where (C1,C2) in (('a', 'a'), ('b', 'b'))
           →    SearchCnd: RANGES(CS-CE)
                       [('a', 'a',MIN),( 'a', 'a',MAX)] ,
                        [('b', 'b',MIN),( 'b', 'b',MAX)]
```

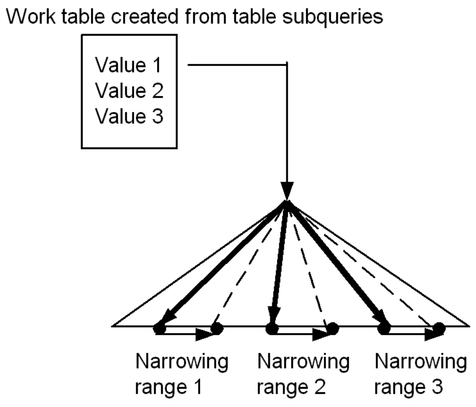
Note: The index component columns to be used are `C1`, `C2`, and `C3`.

RANGES by a set of narrowed ranges

`RANGES(...)` `[`*set-of-narrowing-start-value*`,`*set-of-narrowing-end-value*`]`:

Use this type of `RANGES` to search an index more than once using a set of narrowed ranges (work table created from table subqueries).

An example of a search condition is `C1 =any(select C1 from T2).`



Example 1

```
where C1 =any(select C1 from T2)
```

```
        →    SearchCnd: RANGES(CS-CE)
                    [((SUBQ(2)),MIN),((SUBQ(2)),MAX)]
```

Note: The index component columns to be used are `C1` and `C2`.

Example 2

```
where (C1,C2) = any(select C1, C2 from T2)
```

```
        →    SearchCnd: RANGES(CS-CE)
                  [((SUBQ(2)),(SUBQ(2)),MIN),((SUBQ(2)),
                    (SUBQ(2)),MAX)]
```

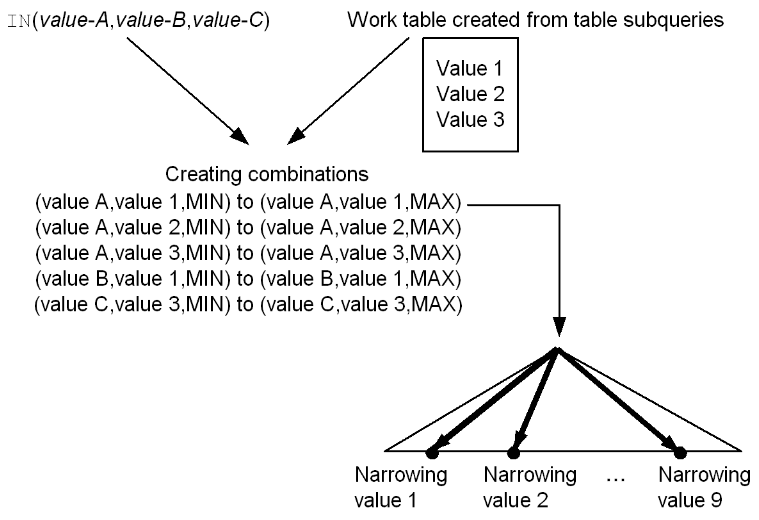
Note: The index component columns to be used are `C1`, `C2`, and `C3`.

RANGES by multiple sets of narrowed ranges

`RANGES(...)`
`[`*narrowing-start-value-or-set-of-narrowing-start-value*`,`*narrowing-end-value-or-set-of-narrowing-end-value*`], ...`
`,[`*narrowing-start-value-or-set-of-narrowing-start-value*`,`*narrowing-end-value-or-set-of-narrowing-end-value*`]:`

Use this type of `RANGES` to search an index more than once using the result of combining sets of narrowed ranges (work table created from table subqueries) and the `IN` predicate.

An example of a search condition is `C1 in ('a','b','c') and C2=any(Select C1 from T2)`.


IN(value-A,value-B,value-C)
Work table created from table subqueries
Value 1
Value 2
Value 3
Creating combinations
(value A,value 1,MIN) to (value A,value 1,MAX)
(value A,value 2,MIN) to (value A,value 2,MAX)
(value A,value 3,MIN) to (value A,value 3,MAX)
(value B,value 1,MIN) to (value B,value 1,MAX)
(value C,value 3,MIN) to (value C,value 3,MAX)
Narrowing value 1
Narrowing value 2
...
Narrowing value 9

Example 1

```
where C1 in ('a','b','c') and C2=any(Select C1 from T2)
```

$\rightarrow$ `SearchCnd: RANGES(CS-CE)`

```
[('a',(SUBQ(2)),MIN),('a',(SUBQ(2)),MAX)],

[('b',(SUBQ(2)),MIN),('b',(SUBQ(2)),MAX)],

[('c',(SUBQ(2)),MIN),('c',(SUBQ(2)),MAX)]
```

Note: The index component columns to be used are `C1`, `C2`, and `C3`.

Example 2

```
where C1 in ('a','b') and
(C2,C3)=any(Select C1,C2 from T2) and C4>'A'
```

$\rightarrow$
```
SearchCnd: RANGES(CS-CE)
               [('a',(SUBQ(2)),(SUBQ(2)),'A'),
               ('a',(SUBQ(2)),(SUBQ(2)),MAX)],
               [('b',(SUBQ(2)),(SUBQ(2)),'A'),
               ('b',(SUBQ(2)),(SUBQ(2)),MAX)]
```

Note: The index component columns to be used are `C1`, `C2`, `C3`, and `C4`.

(3) Values used in a narrowed range

(a) Literal

The utility displays a value if a literal is used to narrow the search condition. For details about the literal output format, see the *HiRDB Version 8 SQL Reference*.

If a LIKE or SIMILAR predicate is used to specify a pattern character string that results in right truncation, the literal is displayed as shown in Table 17-2.

Table 17-2: Display example and search range when LIKE predicate is used to specify a pattern character string that results in right truncation

Data type of column	Pattern character string	Output example of narrowed range based on search condition	Index search range*
Fixed length (e.g., char(5))	Literal	RANGE(CS-CE) ['abc'00,'abc'ff] (where 'abc%' is the literal) For the character string found, eliminate the first '%' part or the part following '_', then fill 0s up to the defined length for the start value and fill fs up to the defined length for the end value.	X'6162630000' to X'616263ffff'
	Embedded variable or ? parameter	RANGE(CS-CE) [<?(1)0>,<?(1)f>] Fill one 0 for the start value and one 'f' for the end value.	X'6162630000' to X'616263ffff' (? parameter is 'abc%'.)
	SQL variable or SQL parameter (SQLVAR assumed as the name of SQL variable)	RANGE(CS-CE) [<SQLVAR0>,<SQLVARf>] Fill one 0 for the start value and one 'f' for the end value.	X'6162630000' to X'616263ffff' (SQL variable is 'abc%'.)
Variable length (e.g., varchar(5))	Literal	RANGE(CS-CE) ['abc','abc'ff] (where 'abc%' is the literal) For the start value, eliminate the first '%' part or the part following '_'; for the end value, fill fs in the start value up to the defined value.	X'616263' to X'616263ffff'
	Embedded variable or ? parameter	RANGE(CS-CE) [<?(1)>,<?(1)f>] Only for the end value, fill one f.	X'616263' to X'616263ffff' (? parameter is 'abc%'.)
	SQL variable or SQL parameter (SQLVAR assumed as the name of SQL variable)	RANGE(CS-CE) [<SQLVAR>,<SQLVARf>] Only for the end value, fill one f.	X'616263' to X'616263ffff' (SQL variable is 'abc%'.)

* Character code is X'61' for a, X'62' for b, and X'63' for c.

(b) ? (?-number)

The utility displays ? (?-*number*) for an embedded variable or ? parameter.

`?`-*number* is a sequence number assigned to each embedded variable or `?` parameter in the SQL statement in the order it appears from left to right, beginning at number 1.

If a `LIKE` or `SIMILAR` predicate is used to specify a pattern character string that results in right truncation, `?` is displayed as shown in Table 17-2.

(c) SQL variable name

The utility displays the name of an SQL variable.

If a `LIKE` or `SIMILAR` predicate is used to specify a pattern character string that results in right truncation, the SQL variable name is displayed as shown in Table 17-2.

(d) SQL parameter name

The utility displays the name of an SQL parameter.

If a column with new value correlation name and a column with old correlation name are used in a trigger SQL statement, the former is displayed as `(NEWROW).`*column-name* and the latter as `(OLDROW).`*column-name*.

If a `LIKE` or `SIMILAR` predicate is used to specify a pattern character string that results in right truncation, the SQL parameter name is displayed as shown in Table 17-2.

(e) USER

The utility displays the user when using `USER` to narrow the search condition.

(f) CURRENT DATE

The utility displays the current date when using `CURRENT DATE` to narrow the search condition.

(g) CURRENT TIME

The utility displays the current time when using `CURRENT TIME` to narrow the search condition.

(h) CURRENT_TIMESTAMP(p)

The utility displays `CURRENT_TIMESTAMP(`*p*`)` when using `CURRENT_TIMESTAMP` to narrow the search condition (*p* = 0, 2, 4, or 6).

(i) SUBQ(subquery-ID)

The utility displays `SUBQ(`*subquery-ID*`)` when using the result of a subquery that does not contain a set operation to narrow the search condition. The parentheses enclose the query ID.

(j) SUBQEX(query-expression-ID)

The utility displays `SUBQEX(`*query-expression-ID*`)` when using the result of a subquery that contains a set operation to narrow the search condition. The parentheses enclose the query ID.

(k) Set function

The utility displays set function when a set function is used to narrow the search condition.

(l) NULL

The utility displays `NULL` when using the null value to narrow the search condition.

(m) table-name.column-name

The utility displays *table-name.column-name* when using columns to narrow the search condition.

If the table name is a correlation name, the correlation name is displayed in parentheses instead.

(n) (NEW ROW).column-name

The utility displays `(NEW ROW).`*column-name* when using an insertion or updating value for a foreign key to narrow the search condition in a subquery that is created internally by HiRDB for checking constraints.

(o) table-name.column-name[subscript]

The utility displays *table-name.column-name*`[`*subscript*`]` when using repetition columns (subscript is an integer) to narrow the search condition.

If the table name is a correlation name, the correlation name is displayed in parentheses instead.

(p) MIN

The utility displays `MIN` to search for the smallest index key value (for a character string, the smallest character encoding).

(q) MAX

The utility displays `MAX` to search for the largest index key value (for a character string, the largest character encoding).

17.5.13 Key conditions

(1) Overview of key conditions

`KeyCnd`: *key-condition*

A key condition makes evaluation possible only with the index component columns.

Given a key condition, the utility can evaluate the search condition only with the index pages without having to reference the data pages, thereby achieving high-speed retrieval. Evaluation based on key conditions is effective if it is conducted after narrowing the index search range by search conditions. If it is impossible to evaluate a predicate with the specified search conditions and key conditions, the utility evaluates it during data page retrieval.

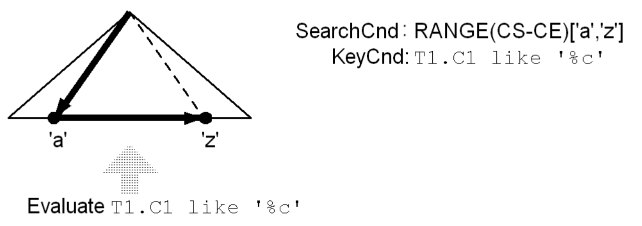
To specify multiple key conditions, use `AND` and `OR` along with parentheses to indicate the priority levels of `AND` and `OR`.

(a) Both search condition and key condition specified

The utility narrows the index search range by the specified search condition, then conducts evaluation based on the specified key condition.

Example

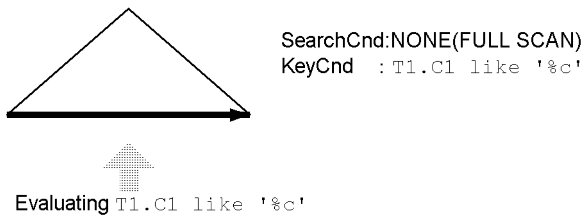
where T1.C1 between 'a' and 'z' and T1.C1 like '%c'



(b) Only key condition specified

The utility conducts evaluation based on the specified key condition from the top to the end of the index. Because the utility needs to search the entire range of index, performance is lower than when both search and key conditions are specified.

Example
```
where T1.C1 like '%c'
```



(c) Key condition deleted during execution

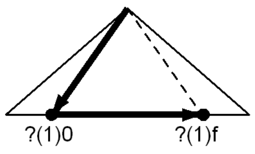
If you specify an embedded variable, `?` parameter, SQL variable, `LIKE` predicate containing an SQL parameter, or a `SIMILAR` predicate in a pattern character string, and the pattern character string given during execution is something such as `abc%d`, the utility searches as the search condition for values that begin with `abc` and determines as the key condition whether or not the value ends with `d`. If the pattern character string given during execution results in right truncation, such as '`abc%`', the utility assumes that the condition can be evaluated simply by searching for a value that begins with '`abc`' and does not evaluate the key condition during execution.

The utility displays in diamond brackets (`< >`) a key condition that can be deleted due to the value of pattern character string.

Example

```
where T1.C1 like ?
```

Where the value of the `?` parameter is '`abc%`' (data type of `C1`: `CHAR(5)`).



```
SearchCnd : RANGE(CS-CE) [<?(1)0>,<?(1)f>]
KeyCnd    : <T1.C1 like ?(1)>
```

Explanation:

- `?`: '`abc%`'

  The utility searches the range from `X'6162630000'` to `X'616263ffff'`. (The key condition is deleted).

- `?`: '`abc%d`'

  The utility searches the range from `X'6162630000'` to `X'616263ffff'` and evaluates `T1.C1 like 'abc%d'` using the key condition.

- `?`: '`%abc`'

  The utility searches the entire range of the index and evaluates `T1.C1 like '%abc'` using the key condition (only the search condition is deleted; for details about the search conditions, see *17.5.12 Search conditions*).

- `?(1)` indicates the `?` number. For details, see (3) *Values used in key conditions*.

(2) Predicates used in key conditions

This section presents the format of predicates that can be included in the key condition. You cannot use a quantified predicate or `EXISTS` predicate in a key condition. The examples of key conditions explained here may be used for search conditions to achieve high-speed retrieval depending on the index definition method.

(a) NULL predicate

value `is null`:

The utility evaluates to see whether the index key value is null.

Example
```
    where T1.C1 is null
        →  KeyCnd: T1.C1 is null
```

value `is not null`:

The utility evaluates to see whether the index key value is not null.

Example
```
    where T1.C1 is not null
        →  KeyCnd: T1.C1 is not null
```

(b) IN predicate

value `in (`*value*`,`*value*`,..., `*value*`)`:

The utility evaluates to see whether the index key value matches any of the specified values.

Example
```
    where T1.C1 in ('a','b','c')
        →  KeyCnd: T1.C1 in ('a','b','c')
```

value `not in (`*value*`,`*value*`,..., `*value*`)`:

The utility evaluates to see whether the index key value matches none of the specified values.

Example
```
    where T1.C1 not in ('a','b','c')
        →  KeyCnd: T1.C1 not in ('a','b','c')
```

(c) LIKE predicate

value `like` *pattern-character-string*`[escape` *escape-character*`]`:

The utility evaluates to see whether the index key value matches the specified pattern.

Example
```
    where T1.C1 like '%a'
        →  KeyCnd: T1.C1 like '%a'
```

value `not like` *pattern-character-string*`[escape` *escape-character*`]`:

The utility evaluates to see whether the index key value does not match the specified pattern.

Example

```
where T1.C1 not like '%a\_' escape '\'
    →  KeyCnd: T1.C1 not like '%a\_' escape '\'
```

(d) XLIKE predicate

value `xlike` *pattern-character-string*`[escape` *escape-character*`]`:

The utility evaluates to see whether the index key value matches the specified pattern regardless of the case (uppercase or lowercase).

Example

```
where T1.C1 xlike '%a'
    →  KeyCnd: T1.C1 xlike '%a'
```

value `not xlike` *pattern-character-string*`[escape` *escape-character*`]`:

The utility evaluates to see whether the index key value does not match the specified pattern regardless of the case (uppercase or lowercase).

Example

```
where T1.C1 not xlike '%a\_' escape '\'
    →  KeyCnd: T1.C1 not xlike '%a\_' escape '\'
```

(e) BETWEEN predicate

value `between` *value-1* `and` *value-2*:

The utility evaluates to see whether the index key value is in the range from *value-1* to *value-2*. Note that `not between` is converted to a combination of a comparison predicate (<, >) and `OR`.

Example 1

```
where T1.C1 >= 'a' and T1.C1 <= 'z'
     →  KeyCnd: T1.C1 between 'a' and 'z'
```

Example 2

```
where T1.C1 not between 'a' and 'z'
     →  KeyCnd: T1.C1 < 'a' OR T1.C1 > 'z'
```

(f) Comparison predicate

value`{=, <, <=, >, >=, <>}`*value*:

The utility evaluates to see whether the index key value meets the comparison predicate (`=, <, <=, >, >=, <>`).

Example 1

```
where T1.C1=T1.C2
```

```
        → KeyCnd: T1.C1=T1.C2
```

Example 2

```
    where T1.C1=(select C1 from T2)
        → KeyCnd: T1.C1=SUBQ(2)
```

(g) Structured repetition predicate

`array(...)[any](`*key-condition*`)`:

From the conditions specified with a structured repetition predicate, the utility eliminates the conditions that can be evaluated by the search conditions, then displays the conditions to be evaluated by the key condition.

Example

```
    where array(T1.C1,T1.C2)[any]
        (T1.C1<'a' or (T1.C1>'z' and T2.C2='a'))
        → KeyCnd: array(...)[any]
               (T1.C1<'a' or (T1.C1>'z' and T2.C2='a'))
```

(h) SIMILAR predicate

value `similar to` *pattern-character-string*`[escape` *escape-character*`]`:

The utility evaluates whether the index key value matches the specified pattern.

```
Example: where T1.C1 similar to '%a'
    → KeyCnd: T1.C1 similar to '%a'
```

value `not similar to` *pattern-character-string*`[escape` *escape-character*`]`:

The utility evaluates whether the index key value does not match the specified pattern.

```
Example: where T1.C1 not similar to '%a\_' escape '\'
    → KeyCnd: T1.C1 not similar to '%a\_' escape '\'
```

(3) Values used in key conditions

(a) table-name.column-name

The utility displays *table-name.column-name* if a column name is specified in the key condition.

If the table name is a correlation name, the correlation name is displayed in parentheses instead.

(b) (NEW ROW).column-name

The utility displays `(NEW ROW).`*column-name* when using an insertion or updating value for a foreign key to narrow the search condition in a subquery that is created internally by HiRDB for checking constraints.

(c) table-name.column-name[subscript]

The utility displays *table-name.column-name*`[`*subscript*`]` if a repetition column (subscript is an integer or `ANY`) is specified in the key condition.

If the table name is a correlation name, the correlation name is displayed in parentheses instead.

(d) literal

The utility displays a value if a literal is specified in the key condition. For details about the literal output format, see the *HiRDB Version 8 SQL Reference*.

(e) ? (?-number)

The utility displays `? (?-`*number*`)` if an embedded variable or `?` parameter is specified in the key condition. `?-`*number* is a sequence number assigned to each embedded variable or `?` parameter in the SQL statement in the order it appears from left to right, beginning at number 1.

(f) SQL variable name

The utility displays the name of an SQL variable if it is specified in the key condition.

(g) SQL parameter name

The utility displays the name of an SQL parameter if it is specified in the key condition.

If a column with new value correlation name and a column with old correlation name are used in a trigger SQL statement, the former is displayed as `(NEWROW).`*column-name* and the latter as `(OLDROW).`*column-name*.

(h) USER

The utility displays the user if `USER` is specified in the key condition.

(i) CURRENT DATE

The utility displays the current date if `CURRENT DATE` is specified in the key condition.

(j) CURRENT TIME

The utility displays the current time if `CURRENT TIME` is specified in the key condition.

(k) CURRENT_TIMESTAMP(p)

The utility displays `CURRENT_TIMESTAMP(`*p*`)` if `CURRENT_TIMESTAMP` is used in the key condition (p = 0, 2, 4, or 6).

(l) (SUBQ(query-ID))

The utility displays `SUBQ(`*query-ID*`)` if a row subquery or a scalar subquery that does not contain a set operation is specified in the key condition. The parentheses enclose the query ID.

(m) SUBQEX(query-expression-ID)

The utility displays `SUBQ(`*query-ID*`)` if a row subquery or a scalar subquery that contains a set operation is specified in the key condition. The parentheses enclose the query ID.

(n) Scalar operation

The utility displays a scalar operation if it is specified in the key condition.

(o) row value constructor

The utility displays a row value constructor if it is specified in the key condition.

17.5.14 Types of ORDER BY processing methods

(1) HiRDB/Single Server

The following processing methods are available to a HiRDB/Single Server:

`SORT CANCEL BY INDEX[(LIMIT SCAN)]`

This method skips the sorting for `ORDER BY` in situations in which the columns subject to `ORDER BY` can be sorted by retrieving their index.

If a function for obtaining the first *n* rows of a search result is used, the utility may display `(LIMIT SCAN)`. In such a case, the utility searches only the first *n* rows of the index and obtains the first *n* rows of the search result.

`SORT CANCEL BY JOIN`

This method skips the sorting for `ORDER BY` in situations in which all columns subject to `ORDER BY` are specified in the join condition because merge-join sorting takes place.

`SORT CANCEL BY DISTINCT[(LIMIT SCAN)]`

This method skips the sorting for `ORDER BY` in situations in which all the value expressions subject to `ORDER BY` are sorted by `DISTINCT` processing.

If a function for obtaining the first *n* rows of a search result is used, the utility may display `(LIMIT SCAN)`. In such a case, the utility obtains the first *n* rows of the `DISTINCT` processing result.

`SORT CANCEL BY GROUPING[(LIMIT SCAN)]`

This method skips the sorting for `ORDER BY` in situations in which all the columns subject to `ORDER BY` can be sorted by grouping.

If a function for obtaining the first *n* rows of search result is used, the utility may display `(LIMIT SCAN)`. In such a case, the utility obtains the first *n* rows of the grouping processing result.

`LIST SORT`

This method obtains the result of `ORDER BY` by creating a work table and sorting it.

`LIMIT SCAN`

The utility may display this if a function for obtaining the first *n* rows of the search result is used. In such a case, the utility obtains the first *n* rows of the `ORDER BY` result by sorting only a limited number of rows, not all the rows that satisfy the search condition.

`PRE-SORT JOIN`

In situations in which all the columns or value expressions subject to `ORDER BY` are included in an outer table before it is joined or in an outer-joined table, this method executes sorting before the join operation, executes the join operation in the sorted order, then obtains the result of `ORDER BY`.

`PRE-SORT SET OPERATION[(LIMIT SCAN)]`

This method executes a sort operation for each query prior to a set operation to use the sorted order for operations. The method then obtains the result of `ORDER BY` and set operation.

If a function for obtaining the first *n* rows of the search result is used, the utility may display `(LIMIT SCAN)`. In such a case, the utility performs the set operation until it obtains the first *n* rows and then obtains the first *n* rows of the result of the `ORDER BY` and the set operation.

`SORT CANCEL FOR SAME VALUES`

This method omits sort processing by `ORDER BY` when `ORDER BY`'s target value expressions are all guaranteed to be the result of window functions and have the same value in all rows.

(2) HiRDB/Parallel Server

The following processing methods are available to a HiRDB/Parallel Server:

`SORT CANCEL BY INDEX[(LIMIT SCAN)]`

This method skips the sorting for `ORDER BY` in situations in which the columns subject to `ORDER BY` can be sorted by retrieving their index.

If a function for obtaining the first *n* rows of the search result is used, the utility may display `(LIMIT SCAN)`. In such a case, the utility searches only the first *n* rows of the index and obtains the first *n* rows of the search result.

`SORT CANCEL BY DISTINCT[(LIMIT SCAN)]`

This method skips the sorting for `ORDER BY` in situations in which all the value expressions subject to `ORDER BY` are sorted by `DISTINCT` processing.

If a function for obtaining the first *n* rows of the search result is used, the utility may display `(LIMIT SCAN)`. In such a case, the utility obtains the first *n* rows of the `DISTINCT` processing result.

`SORT CANCEL BY GROUPING[(LIMIT SCAN)]`

This method skips the sorting for `ORDER BY` in situations in which all the columns subject to `ORDER BY` can be sorted by grouping.

If a function for obtaining the first *n* rows of the search result is used, the utility may display `(LIMIT SCAN)`. In such a case, the utility obtains the first *n* rows of the grouping processing result.

`LIST SORT`

This method obtains the result of `ORDER BY` by creating a work table and sorting it at each back-end server.

`LIMIT SCAN`

The utility may display this if a function for obtaining the first *n* rows of the search result is used. In such a case, the utility obtains the first *n* rows of the `ORDER BY` result by sorting only a limited number of rows, not all the rows that satisfy the search condition.

`PRE-SORT JOIN`

In situations in which all the columns or value expressions subject to `ORDER BY` are included in an outer table before it is joined or in an outer-joined table, this method executes sorting before join operation, executes join operation in the sorted order, then obtains the result of `ORDER BY`.

`FLOATABLE SORT`

This method obtains the result of `ORDER BY` by transferring data to multiple floating servers, creating a work table, and sorting the data in it.

`LIST SORT WITH SET OPERATION[(LIMIT SCAN)]`

This method executes the sorting for `ORDER BY` during a set operation without executing a sort operation for each query. The method then executes operation in the sorted order to obtain the result of `ORDER BY` and set operation.

If a function for obtaining the first *n* rows of the search result is used, the utility may display `(LIMIT SCAN)`. In such a case, the utility performs the set operation until it obtains the first *n* rows and then obtains the first *n* rows of the result of the `ORDER BY` and the set operation.

`SORT CANCEL FOR SAME VALUES`

This method omits sort processing by `ORDER BY` when `ORDER BY`'s target value expressions are all guaranteed to be the result of window functions and have the same value in all rows.

17.6 Concept of tuning

You can improve retrieval performance by modifying the SQL statements and index definitions on the basis of `pdvwopt`'s output result.

Modifying the access path may not always result in improved performance, depending on the data characteristics. When modifying the access path, be sure to evaluate the performance.

For details about performance design with regard to accesses to foreign tables, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

17.6.1 Examples of tuning procedure

This section shows a sample tuning procedure based on `pdvwopt`'s output result. Use the information provided in Sections *17.6.2 Notes about index definitions* to *17.6.4 Notes about join retrieval* as guidelines for tuning.

```
*****     Result of SQL Optimizer     *****
Version         : HiRDB/Parallel Server VV-RR 4BES
UAP Name        : pdsql   ------------------------> Is the UAP name correct?
UAP Source      : pdsql-21775
Authorization   : user1   ------------------------> Is the authorization identifier correct?
--------------------------------------------------------------------------------
Section No      : 1
Optimize Time   : 2001-01-07 10:59:16.591727  ---> Is the SQL statement execution time correct?
Optimize Mode   : COST_BASE_2
SQL Opt Level   : 0x00000420(1056) = "PRIOR_NEST_JOIN"(32),"RAPID_GROUPING"(1024)
Add Opt Level   : 0x00000003(3) = "COST_BASE_2"(1),"APPLY_HASH_JOIN"(2)
  ------------> Are the SQL optimization option and extended SQL optimization option optimum?1
SQL             : select T1.C1,T1.C2 from T1,T2,T3
                      where T1.C1='a' and T1.C2 like '%A'
                        and T1.C1=T2.C1 and T1.C2=T2.C2 and T2.C3>=1995
                        and T1.C1=T3.C1 and T1.C2=T3.C2 order by T1.C2
                                                   --> Is the SQL statement correct?
Work Table      : 3
Total Cost      : 3314.567244
----- QUERY ID : 1 -----
Query Type      : QUERY LIST(SORT) {LLID1(2)}
Order by Mode   : FLOATABLE SORT
FLT Server      : 2 (ORDER BY)  2-CLM 1TO1
JOIN
 # Join  ID     : 1   ----------------------------> Are the tables joined in the expected order?
   L Table      :  T1
   R Table      :  T2
   Join Type    : 2-CLM NESTED LOOPS JOIN(INNER) BROADCAST(FROM L TO R)
 -------------> Is it possible to change the inner table's partitioning key to something other than BROADCAST?2
   FLT Server   : 0
 # Join  ID     : 2
   L Table      :  T3
   R Join ID    :  1 LIST{JRLST(1)}
   Join Type    : 2-CLM HASH JOIN(INNER) 1-CLM HASH(FROM R TO L)
 -------------> Is this the expected join method?3
   FLT Server   : 0
```

```
SCAN
 # Table Name  : T1
   Cost        : N (10000000ROW) {T-2962.358541,I-763.682244,AND-766.202330}
   RDAREA      : NON DIVISION (1RD/1BES) [0x07(7)] ALL
   Scan Type   : MULTI COLUMNS KEY SCAN
   Index Name  : X1 (2) (+C2,+C1)   --------------> Is this the expected index?
                     SearchCnd : RANGE(CS-CE) [(MIN,'a    '),(MAX,'a    ')] (FULL SCAN)
                      -----------------------------> Is the search range narrowed?4
                      KeyCnd    : T1.C1='a    ' AND T1.C2 like '%A'
 # Table Name  : T2
   Cost        : N (20000000ROW) {T-1388.652368,I-408.252884,AND-2647.479838}
   RDAREA      : 1-CLM FIX HASH(HASH6) (2RD/2BES) [0x07(7),0x0d(13)] ALL
   Scan Type   : MULTI COLUMNS KEY SCAN
   Index Name  : X2 (3) (+C1,+C2,+C3)
                      SearchCnd : RANGE(CS-CE) [(T1.C1,T1.C2,1995),(T1.C1,T1.C2,MAX)]
 # Table Name  : T3
   Cost        : N (20000000ROW) {T-1621.172368}
  RDAREA      : 1-CLM FIX HASH(HASH6)(4RD/2BES)[0x07(7),0x08(8),0x0d(13),0x0e(14)]ALL
   Scan Type   : TABLE SCAN   ------------------> Is this the expected scan type?
```

Note

This output result is partially edited; it is not the actual display example.

1 Check to see if the specified SQL optimization option and extended SQL optimization option are optimum to the database status. For details about the SQL optimization options and extended SQL optimization options, see the *HiRDB Version 8 UAP Development Guide*.

2 If the transfer method is `BROADCAST`, `KEY RANGE PARTIAL BROADCAST`, or `PARTIAL BROADCAST` during the nested loop join, performance may improve if the inner table's partitioning key subject to nested loop join is changed to something other than `BROADCAST`, `KEY RANGE PARTIAL BROADCAST`, or `PARTIAL BROADCAST`.

For details, see *17.6.4(1) Nested loop join using a partitioning key*.

Before change:

```
 # Join  ID    : 1
   L Table     :  T1
   R Table     :  T2
   Join Type   : 2-CLM NESTED LOOPS JOIN(INNER) BROADCAST(FROM L TO R)
   FLT Server  : 0
```

↓ Changing table `T2`'s partitioning key to `C1` (join key).

After change:

```
# Join  ID    : 1
  L Table     :  T1
  R Table     :  T2
  Join Type   : 2-CLM NESTED LOOPS JOIN(INNER) 1-CLM HASH(FROM L TO R)
  FLT Server  : 0
```

Use the column with the most even data distribution as the partitioning key. If there is no such column or multiple columns are joined, select more than one joined column as the partitioning key in such a manner that data becomes fairly even.

3 If the join result of `T1` and `T2` is narrowed down, you can improve the retrieval speed by applying a nested loop join to `T3` and the join result of `T1` and `T2`. Define multicolumn index `T3` that contains all columns of `T3` (`C1`, `C2`) from the join condition including the columns of `T3` (`T1.C1=T3.C1 and T1.C2=T3.C2`). For details, see *17.6.2(2) Index definitions for a table used for join retrieval*.

Before change:

```
# Join  ID    : 2
  L Table     :  T3
  R Join ID   :  1 LIST{JRLST(1)}
  Join Type   : 2-CLM HASH JOIN(INNER) 1-CLM HASH(FROM R TO L)
  FLT Server  : 0
                :
                :
                :
# Table Name  : T3
  Cost        : N (2000000000ROW)  {T-1621.172368}
  RDAREA      : 1-CLM FIX HASH (HASH6)  (4RD/2BES) [...] ALL
  Scan Type   : TABLE SCAN
```

↓ Defining an index for `T3(C1, C2)`

After change:

```
# Join  ID     : 2
  L Join ID    :  1
  R Table      :  T3
  Join Type    : 2-CLM NESTED LOOPS JOIN(INNER) 1-CLM HASH(FROM L TO R)
  FLT Server   : 0
                 :
                 :
                 :
# Table Name   : T3
  Cost         : N (2000000000ROW)  {T-1621.172368,T-1581.852884}
  RDAREA       : 1-CLM FIX HASH (HASH6)  (4RD/2BES) [...] ALL
  Scan Type    : MULTI COLUMNS KEY SCAN
  Index Name   : X3 (2D) (+C1,+C2)
                    SearchCnd : AT [(T1.C1,T1,C2)]
```

4 The utility searches the entire range of index because it cannot form the range of conditions for the first index component column. The utility reverses index component columns 1 and 2 because = is specified for the second index component column. For details, see *17.6.2(1) Index definitions*.

Before change:

```
Index Name  : X1 (2) (+C2,+C1)
                  SearchCnd : RANGE(CS-CE) [(MIN,'a    '),(MAX,'a    ')] (FULL SCAN)
                   KeyCnd    : T1.C1='a    ' AND T1.C2 like '%A'
```

↓ Deleting the index for `T1(C2, C1)` and defining an index for `T3(C1,C2)`

After change:

```
Index Name  : X1 (2) (+C1,+C2)
                   SearchCnd : RANGE(CS-CE) [('a    ',MIN),('a    '),MAX]
                   KeyCnd    : T1.C2 like '%A'
```

17.6.2 Notes about index definitions

Use the interactive SQL execution utility (HiRDB SQL Executer) to check the index definition information.

HiRDB SQL Executer available

To obtain a list of defined indexes, use HiRDB SQL Executer's `INDEXES` command. To obtain the names of index component columns from the index name, use the `INDEXCLM` command. For details, see HiRDB SQL Executer's *README*.

HiRDB SQL Executer not available

If HiRDB SQL Executer is not available, obtain the information by searching the data dictionary tables. Examples of searching the data dictionary tables are shown as follows. For details about the data dictionary tables, see the *HiRDB Version 8 UAP Development Guide*.

1. To obtain a list of defined indexes, retrieve `SQL_INDEXES` from the data dictionary tables:

```
SELECT INDEX_ID , INDEX_NAME , TABLE_NAME ,
        UNIQUE_TYPE , VALUE(ARRAY_TYPE,' ') , COLUMN_COUNT
  FROM MASTER.SQL_INDEXES
  WHERE TABLE_SCHEMA LIKE USER
  ORDER BY 1
```

2. To obtain index component columns from the index name, execute join retrieval of `SQL_INDEX_COLINF` and `SQL_COLUMNS` on the data dictionary tables:

```
SELECT Y.INDEX_ORDER , X.COLUMN_ID , X.COLUMN_NAME ,
         VALUE(X.MAX_ELM,1) , Y.ASC_DESC
  FROM MASTER.SQL_COLUMNS X , MASTER.SQL_INDEX_COLINF Y
  WHERE X.TABLE_SCHEMA = Y.TABLE_SCHEMA
    AND X.TABLE_NAME = Y.TABLE_NAME
    AND X.COLUMN_NAME = Y.COLUMN_NAME
    AND Y.INDEX_NAME LIKE 'IDX1'
    AND Y.TABLE_SCHEMA LIKE USER
  ORDER BY 1
```

(1) Index definitions

(a) Checking the display

1. Check the `Scan Type` information. If the performance is poor with `TABLE SCAN` or `AND PLURAL INDEXES SCAN`, consider defining a multicolumn index that includes all columns for which the index is defined.
2. Check the `Index Name` information to see if an expected index is used.
3. Check to see if the range of index is narrowed efficiently by the search condition and key condition. Following are some examples in which the index search range is not narrowed efficiently:
    - There is no search condition (`SearchCnd:NONE(FULL SCAN)` is displayed for the search condition).
    - The range of the first component column narrowed by the search condition is from `MIN` to `MAX (FULL SCAN)` is displayed after the narrowed range of search condition).
    - A wide range of index is being searched.

(b) Better method

If a table is subject to frequent retrieval processing specifying multiple predicates using `AND`, define a multicolumn index consisting of the predicate columns. In this case, specify the columns that are frequently combined with the = predicate as the index component columns in the ascending order of duplicate values contained in the columns.

(c) Reason

The search time is reduced because the index search range is narrowed.

(d) Example

```
SELECT * FROM T1 WHERE C1=10 AND C2=30 AND C3 BETWEEN 10 AND 20
SELECT * FROM T1 WHERE C1=10 AND C2=30 AND C3 <= 15
SELECT * FROM T1 WHERE C1=20 AND C2=40 AND C4 = 60
```

↓

Defines an index of `T1(C1,C2,C3), T1(C1,C2,C4)`.

(e) Notes

1. If there are too many index component columns, the time required to search the index itself increases.
2. If a column is updated, maintenance occurs on the index. If a column is updated frequently, you should not include this column in the index component columns.
3. Try to minimize the number of indexes to be defined.

(2) Index definitions for a table used for join retrieval

(a) Checking the display

If the `Join Type` is sort merge join and the performance is poor because a large amount of data is sorted, but the table retrieval condition can be narrowed efficiently, you can achieve effective retrieval by creating an index for the inner table's joined columns and using a nested loop join.

(b) Better method

For a table used for join retrieval with another table with a search range that is narrowed by limitations, define an index with a first component column of the joined column.

If multiple join conditions (*n*) are specified frequently, define an index that consists of the joined columns as its component columns 1 through *n*. If there are search conditions other than the join conditions, specify those search conditions following the joined columns (after component columns *n* + 1) to define the index.

(c) Reason

Because a nested loop join is executed on the outer table for which limitations are specified and the inner table, index for which is defined for the joined column, the utility can use the index for a join operation, thereby reducing the number of input/output operations.

(d) Example

```
SELECT * FROM T1,T2
WHERE T1.C3=20 AND T1.C1=T2.C1 AND T1.C2=T2.C2 AND T2.C3>10
                   JOINED COLUMN OF INNER TABLE IS T2 (C1, C2)
```

↓

Defines an index of `T2(C1,C2,C3).`

(e) Notes

1. If there is no limitation on the outer table in the join conditions, a nested loop join usually does not take place.
2. If limitations are specified for both inner and outer tables, a nest loop join may not take place.
3. To use a nested loop join preferentially or forcibly, specify SQL optimization option *forced nest-loop-join* or *prioritized nest-loop-join*. For details, see the *HiRDB Version 8 UAP Development Guide*.

(3) Index used for outer join

(a) Checking the display

Check to see if an expected index is used for outer join (`LEFT OUTER JOIN`).

(b) Better method

None of the following indexes is used for outer join:

- Index for the outer table's column that is specified for the `ON` retrieval condition in the `FROM` clause
- Index for the inner table's column that is specified in the `WHERE` clause

To execute retrieval using an index, evaluate whether the conditions for the outer table can be specified in the `WHERE` clause and the conditions for the inner table in the `ON` retrieval condition.

Note

The following shows an outer table and an inner table for outer join:

```
outer-table LEFT OUTER JOIN inner-table ON ...
                  WHERE ...
```

(c) Reason

The utility retrieves all columns from the outer table whether or not the `ON` retrieval condition in the `FROM` clause is true. Therefore, the limitation on the outer table's columns specified in the `ON` retrieval condition in the `FROM` clause are not subject to the narrowing of range using indexes.

For the inner table's columns, the utility evaluates the `ON` retrieval condition in the `FROM` clause, fills the null value, then evaluates the condition in the `WHERE` clause. Therefore, the limitation on the inner table's columns specified in the `WHERE` clause is not subject to the narrowing of range using indexes.

(d) Example

```
SELECT * FROM T1 LEFT JOIN T2 ON T1.C1=T2.C1 and T1.C2='a'
         and T2.C2='b'
  WHERE T1.C3='c' and T2.C4='d'
```

The underlined columns are evaluated using the index.

(4) Index definitions for a table used for retrieval specifying ORDER BY or GROUP BY

(a) Checking the display

If executing a sort operation for `ORDER BY` processing to retrieve a single table, you can eliminate the sort operation by modifying the index definitions in such a manner that the existing index's ascending or descending order is used to sort the `ORDER BY` column.

(b) Better method

If you are retrieving a single table for which `ORDER BY` or `GROUP BY` is specified after narrowing the range by the = predicate (*column-specification=value-specification*) or `IS NULL` predicate (*column-specification* `IS NULL`), define an index that meets the following two conditions:

- The column specification in the = or `IS NULL` predicate includes component columns 1 through *n* consecutively.
- The columns specified in `GROUP BY` or `ORDER BY` consecutively follow the component column *n* + 1.

(c) Reason

The CPU time and input/output time decrease because the range of data to be accessed is narrowed by using an index, and the sort operation can be omitted.

(d) Example

```
SELECT C2, C3 FROM T1
WHERE C1=10 AND C2= ? AND C3>10
ORDER BY C4 DESC,C5 ASC
```

↓

Defines an index of `T1(C1 ASC,C2 ASC,C4 DESC,C5 ASC)` or `T1(C1 ASC,C2 ASC,C4 DESC,C5 ASC,C3 ASC)`.

(e) Notes

Sort operation cannot be eliminated if any of the following conditions is met:

Limitations common to both a HiRDB/Single Server and a HiRDB/Parallel Server

- `DISTINCT` is specified.
- Specified retrieval condition selects a retrieval method that does not use an index.
- `T1` is partitioned and a non-partitioning key index is partitioned within the same server (index with an `Index Name` attribute of `d`). This does not apply when *column=value-specification* is specified for all partitioning keys.

Limitations to a HiRDB/Single Server

The sort operation cannot be eliminated if `T1` is partitioned and the partitioning keys specified in `CREATE TABLE` are included in the same order of the index component columns from the top (index with an `Index Name` attribute of `d`). This does not apply when *column=value-specification* is specified for all partitioning keys.

Limitations to a HiRDB/Parallel Server

- SQL executes the `INSERT` or `SELECT` statement.
- Specification is made within a subquery.
- A set operation is specified.
- The `HAVING` clause is specified.
- `FOR UPDATE` or `FOR READ ONLY` is specified.
- The sort operation cannot be eliminated if `T1` is partitioned, the partitioning keys specified in `CREATE TABLE` are included in the same order of the index component columns from the top (index with an `Index Name` attribute of `d`), and multiple RDAREAs at the same back-end server are allocated. This does not apply when *column=value-specification* is specified for all partitioning keys.

(5) Index definitions for a table being retrieved specifying set functions MIN and MAX

(a) Checking the display

If `GROUP BY` is not specified and the performance is poor during retrieval specifying set functions `MIN(`*column*`)` or `MAX(`*column*`)` in a selection expression, define an index in the column specification of the set function, so that the value of the set function can be obtained simply by referencing the minimum or maximum index value. In this case, `Group by Mode` is `IMPLICIT MIN-MAX INDEX`.

(b) Better method

There are two better methods:

Retrieval condition not specified

For a table used for retrieval specifying at least one of the set functions `MIN` or `MAX`, define an index that specifies `MIN` or `MAX` for its first component column.

= predicate (or IS NULL predicate) specified as retrieval condition

For a table used for retrieval specifying set function `MIN` or `MAX` after narrowing the range by the retrieval conditions (*n* conditions) specified in the = predicate (or `IS NULL` predicate), define an index that meets all the following conditions:

- The = predicate consecutively specifies component columns 1 through *n*.
- The column specified with `MIN` or `MAX` is component column *n* + 1.
- A column specified in the other retrieval condition is component column *n* + 2 or a subsequent column.

(c) Reason

The CPU time and input/output time decrease because the result is obtained by referencing the minimum or maximum index value for the `MIN` or `MAX` set function, respectively.

(d) Example

Retrieval condition not specified

```
SELECT MIN(C1), MAX(C1) FROM T1
```

↓

Defines an index of `T1(C1)`.

Retrieval condition specified with the = predicate (or IS NULL predicate)

```
SELECT MIN(C1), MAX(C1) FROM T1 WHERE C2=10 AND C3=20 AND C4<30
```

↓

Defines an index of `T1(C2,C3,C1,C4)`.

(e) Notes

The index is not treated as `IMPLICIT MIN-MAX INDEX` if any of the following conditions is met:

Limitations common to both a HiRDB/Single Server and a HiRDB/Parallel Server

- A set function other than `MIN` or `MAX` is specified.
- When there are multiple `MIN`s or `MAX`s, columns in the set function arguments do not match.
- The `GROUP BY` clause is specified.
- The search condition contains a system-defined scalar function, function call, or `IS_USER_CONTAINED_IN_HDS_GROUP`.
- The search condition contains a column that is not an index component column.
- If `FLAT` is specified as a argument of the `MAX` or `MIN` set function, the search condition contains a repetition column.
- There is no index applicable to section (b), previously.
- `T1` is partitioned and retrieval uses a non-partitioning key index that is partitioned within the same server (index with an `Index Name` attribute of `d`). This does not apply when *column=value-specification* is specified for all partitioning keys.

Limitations to a HiRDB/Single Server

- The sort operation cannot be eliminated if `T1` is partitioned and the partitioning keys specified in `CREATE TABLE` are included in the same order of the index component columns from the top (index with an `Index Name` attribute of `d`). This does not apply when *column=value-specification* is specified for all partitioning keys.
- When `T1` is partitioned, retrieval is executed using the index that includes the partitioning keys specified in `CREATE TABLE` in the same order of the index component columns from the top (index with an `Index Name` attribute of `d`). This does not apply when *column=value-specification* is specified for all partitioning keys.

Limitations to a HiRDB/Parallel Server

- A set operation is specified.
- SQL executes the `INSERT` or `SELECT` statement.

- The `HAVING` clause is specified.
- `FOR READ ONLY` is specified.
- When `T1` is partitioned, retrieval is executed using the index that includes the partitioning keys specified in `CREATE TABLE` in the same order of the index component columns from the top (index with an `Index Name` attribute of `d`), and multiple RDAREAs at the same back-end server are allocated. This does not apply when *column=value-specification* is specified for all partitioning keys.

(6) Index definition using the null value as an exception value

(a) Checking the display

Check to see if any of the following is true:

- A retrieval with search condition type `IS NULL` is executed on an index component column that has many duplicated null values, resulting in poor performance.
- Insertion of the null value, updating to the null value, updating from the null value to another value, or deletion of a null-value row takes place, resulting in poor performance.

(b) Better method

Define an index that uses the null value as an exception value.

(c) Reason

If an index is defined that uses the null value as an exception value, the utility creates the index excluding the null value. If `IS NULL` is specified in the retrieval condition, the utility uses `TABLE SCAN` without using the index. However, if a predicate other than `IS NULL` is used (such as = or `between`), the utility uses the index.

If there are many occurrences of null value and the index is used to access data pages at random, you can achieve high-speed retrieval by defining an index that uses the null value as an exception value and using `TABLE SCAN`.

You can reduce the overhead of index maintenance as well as the amount of log information by excluding the null value from the index key values.

(d) Example

`SELECT * FROM T1 WHERE C1 IS NULL`

↓

Defines an index of `T1(C1)` that uses the null value as an exception value.

`SELECT * FROM T1 WHERE C1 IS NULL (` → *index not used*`)`

`SELECT * FROM T1 WHERE C1 ='a' (` → *index used*`)`

(7) Other

- If a lock release wait occurs frequently on a table with a retrieval method of `TABLE SCAN`, you may be able to reduce the occurrences of lock release wait by defining an index for a column specified in the retrieval condition because the range of data to be accessed can be narrowed.
- If a lock release wait occurs frequently during retrieval using an index, evaluate whether or not you can specify a "search using condition evaluation with no lock" that locks only those rows or key values satisfying the retrieval condition. For details about the search using condition evaluation with no lock, see the *HiRDB Version 8 UAP Development Guide*.
- If the retrieval method is `AND PLURAL INDEXES SCAN`, if multiple users update the same table at the same time, deadlock may result. In this case, consider changing the `pd_work_table_option` operand in the system definitions. For details about the `pd_work_table_option` operand, see the *HiRDB Version 8 System Definition*.
- If the access path display utility finds an unused index, you should delete the index. This unneeded index may require extra workload during update processing or extra space in the database.

17.6.3 Notes about index retrieval

(1) Retrieval specifying OR in the retrieval condition (1)

(a) Checking the display

For a single-table retrieval, check to see if `TABLE SCAN` is used, resulting in poor performance.

(b) Better method

If the same conditions are specified on both sides of `OR`, take the condition outside `OR`.

(c) Reason

Single-table retrieval

If an index is defined for the predicate that was taken outside, the search range can be narrowed in the index, thereby reducing the search time.

Join retrieval

If join conditions are connected with `OR`, obtain the direct product (`CROSS JOIN`), then evaluate the join condition for the resulting direct product. If the predicate taken out is a join condition, the utility executes nested loop join or merge join, thereby reducing the number of data match operations during join processing. If the joined column has an index defined, the utility can use the index during join processing, thereby reducing the number of input/output operations.

(d) Examples

OR should be converted to the IN predicate in some cases. For details, see *(2) Retrieval specifying OR in the retrieval condition (2)*.

Single-table retrieval

Index defined for C1

```
SELECT * FROM T1 WHERE (C1=10 AND C2=20) OR (C1=10 AND C2=30)
                ↓
SELECT * FROM T1 WHERE C1=10 AND (C2=20 OR C2=30)
```

If a multicolumn index is defined for (C1,C2) because C1=10 cannot be narrowed sufficiently, performance may be better if C1=10 is kept inside.

Join retrieval

```
SELECT * FROM T1, T2
   WHERE (T1.C1=T2.C1 AND T1.C2=10) OR (T1.C1=T2.C1 AND
T2.C2=20)
                  ↓
SELECT * FROM T1, T2 WHERE T1.C1=T2.C1 AND (T1.C2=10 OR
T2.C2=20)
```

(2) Retrieval specifying OR in the retrieval condition (2)

(a) Checking the display

If the condition is specified with OR for the same column, check to see if performance is poor because there is no search condition or valid search condition to narrow the index search condition.

(b) Better method

The better method depends on whether the HiRDB version is 06-02 or later, the HiRDB version is earlier than 06-01 and a single-column index is defined, or the HiRDB version is earlier than 06-01 and a multicolumn index is defined.

- When the HiRDB version is 06-02 or later

  Use the IN predicate if the = predicate is specified for the same column on both sides of the OR operator and an index is defined for the column.

- When the HiRDB version is earlier than 06-01 and a single-column index is defined

  Use the IN predicate if the = predicate is specified for the same column on both sides of the OR operator and a single-column index is defined for the column.

- When the HiRDB version is earlier than 06-01 and a multicolumn index is defined

  Use the OR operator if the IN predicate is specified for the column which is the first component column of the multicolumn index.

(c) Reason

- When the HiRDB version is 06-02 or later

  For a single-column index or multicolumn index, the utility searches the index for each value specification in the `IN` predicate (search condition type `ATS` or `RANGES`), narrowing the index search range; therefore, the CPU time and the number of input/output operations can be reduced.

- When the HiRDB version is earlier than 06-01 and a single-column index is defined

  For a single-column index, the utility searches the index for each value specification in the `IN` predicate (search condition type `ATS`), narrowing the index search range; therefore, the CPU time and the number of input/output operations can be reduced.

- When the HiRDB version is earlier than 06-01 and a multicolumn index is defined

  For a multicolumn index, the utility searches the minimum value to the maximum value specified in the `IN` predicate (if the value specification includes a non-literal value, the utility references all leaf pages); therefore, the index search range is wide.

  If it is converted to an `OR` operation, `Scan Type` may be `OR PLURAL INDEXES SCAN`. In such a case, the utility searches the index for the left-hand term of the `OR` operator separately from the right-hand term; therefore, the index search range is narrowed and the input/output time is reduced.

(d) Examples

- When a single-column index is defined for `T1(C1)` or a multicolumn index is defined for `T1(C1,C2)` (when the HiRDB version is 06-02 or later)

```
SELECT * FROM T1 WHERE C1=10 OR C1=20
             ↓
SELECT * FROM T1 WHERE C1 IN (10,20)
```

- When a single-column index is defined for `T1(C1)` (when the HiRDB version is earlier than 06-01)

```
SELECT * FROM T1 WHERE C1=10 OR C1=20
             ↓
SELECT * FROM T1 WHERE C1 IN (10,20)
```

- When a multicolumn index is defined for `T1(C1,C2)` (`C1` is the first component column) (when the HiRDB version is earlier than 06-01)

```
SELECT * FROM T1 WHERE C1 IN (10,20)
              ↓
SELECT * FROM T1 WHERE C1=10 OR C1=20
```

(e) Notes

If the following two conditions are satisfied, use the `IN` predicate even for a multicolumn index whose HiRDB version is earlier than 06-01:

- The item specification in the `IN` predicate consists of literals only.
- The item specification in the `IN` predicate contains only a few key values from minimum to maximum.

(3) Join retrieval specifying OR in the join condition

(a) Checking the display

If `Join Type` is `CROSS JOIN`, internal direct product processing is taking place. If an `OR` operator is used in the join condition, you may be able to achieve efficient retrieval by replacing it with equivalent SQL using the set operations separately, specifying both sides of `OR` (`UNION` or `UNION ALL`).

(b) Better method

If join conditions are connected with `OR`, use the set operations separately, specifying both sides of `OR`.

DISTINCT specified in selection expression

Use `UNION`.

DISTINCT not specified in selection expression

Use `UNION ALL`.

(c) Reason

If join conditions are connected with `OR`, the utility obtains the direct product of the tables to be joined, then evaluate the join conditions for the resulting direct product.

If the `OR` operator is separated, the utility executes nested loop join or merge join, thereby reducing the number of data match operations during join processing.

If nested loop join is executed, the utility can use the index during join processing, thereby reducing the number of input/output operations.

(d) Examples

DISTINCT specified

```
SELECT DISTINCT * FROM T1, T2
    WHERE (T1.C1=T2.C1 OR T1.C2=T2.C2) AND T1.C3=10
```

```
          ↓
SELECT * FROM T1, T2 WHERE T1.C1=T2.C1 AND T1.C3=10
  UNION
SELECT * FROM T1, T2 WHERE T1.C2=T2.C2 AND T1.C3=10
```

DISTINCT not specified

```
SELECT * FROM T1, T2 WHERE (T1.C1=T2.C1 OR T1.C2=T2.C2) AND
T1.C3=10
          ↓
SELECT * FROM T1, T2 WHERE T1.C1=T2.C1 AND T1.C3=10
  UNION ALL
SELECT * FROM T1, T2 WHERE T1.C2=T2.C2
   AND (T1.C1<>T2.C1 OR T1.C1 IS NULL OR T2.C1 IS NULL) AND
T1.C3=10
```

(e) Notes

1. If `UNION` is specified, the utility internally creates a work table based on the result of each query specification. In this case, the utility executes a sort operation to eliminate the duplicate values.

   If `UNION ALL` is specified, the utility also internally creates a work table based on the result of each query specification. In this case, however, the utility does not execute a sort operation.

2. If the range of the outer table is not narrowed by limitations, the utility does not use an index during join processing, even when the index is defined for the column subject to join processing. The table for which an index is defined for a column subject to join processing is the inner table, and the other table is the outer table.

3. For the example for `DISTINCT` not specified in (d) previously, the `IS NULL` predicate is not needed if `T1.C1` and `T1.C2` are `NOT NULL` columns or the null value is not inserted. When `DISTINCT` is specified in the selection expression, but an expected result is not obtained, use `UNION` for the set operation, not `UNION ALL`.

(4) Retrieval with a range predicate

(a) Better method

If an `AND` operation joins the >= and <= predicates for the same column, the utility converts it to the `BETWEEN` predicate prior to SQL optimization processing. In this case, the utility converts predicates sequentially beginning with the first one specified in the `WHERE` clause.

Inside HiRDB, the condition joining the >= and <= predicates for the same column in the `AND` operation is treated as being equal to the `BETWEEN` predicate; therefore, the result is the same, whichever is specified.

Check to see if the >= and <= predicates form a range.

(b) Examples

Two different SQL statements selecting the same access path

```
SELECT * FROM T1 WHERE C1 BETWEEN 10 AND 20
SELECT * FROM T1 WHERE C1 >=10 AND C1 <= 20
```

HiRDB's internal conversion

```
SELECT * FROM T1 WHERE C1<=50 AND C1>=20 AND C1<=30
                 ↓
SELECT * FROM T1 WHERE C1 BETWEEN 20 AND 50 AND C1<=30
```

(c) Notes

A redundant predicate results in unneeded condition checking during database access. You should delete any redundant predicate.

(5) Retrieval specifying GROUP BY

(a) Better method

If you are executing `GROUP BY` retrieval on multiple columns at a HiRDB/Parallel Server, specify the grouping column with fewer duplicate values in the `GROUP BY` clause. Note that this is not applicable when 1024 is specified in the SQL optimization option.

(b) Reason

When executing `GROUP BY` processing using multiple floating servers at a HiRDB/Parallel Server, the utility executes hashing on the first column in the `GROUP BY` clause to distribute data to the floating servers. If the first column in the `GROUP BY` clause has only a few duplicate values, the utility can hash evenly to each floating server, thereby improving the effects of parallel processing.

(c) Example

```
SELECT C1,C2,SUM(C3) FROM T1 GROUP BY C1,C2
```

If (`C1`'s duplicates count) > (`C2`'s duplicates count), the utility changes this as follows:

```
SELECT C1,C2,SUM(C3) FROM T1 GROUP BY C2,C1
```

(d) Note

If `1024` is specified in the SQL optimization option, the utility uses all grouping columns for hashing; therefore, there is no need to change the `GROUP BY` order. However, if *17.6.2(4) Index definitions for a table used for retrieval specifying ORDER BY or GROUP BY* applies, do not change the order of grouping columns.

(6) Retrieval specifying both GROUP BY and ORDER BY

(a) Better method

There is an index that includes all GROUP BY columns as its component columns (for a HiRDB/Single Server)

Specify the `GROUP BY` columns in the order of the index component columns.

There is no such index that includes all GROUP BY columns as its component columns

If all `ORDER BY` columns are included in the `GROUP BY` columns during retrieval specifying both `ORDER BY` and `GROUP BY`, specify the `GROUP BY` columns in the order of the `ORDER BY` columns.

(b) Reason

There is an index that includes all GROUP BY columns as its component columns (for a HiRDB/Single Server)

The sort operation can be omitted for `GROUP BY` processing. In this case, the utility still executes a sort operation for `ORDER BY` processing, but retrieval performance improves because the sort operation is executed on the grouped data (fewer data items).

There is no such index that includes all GROUP BY columns as its component columns

If the order of columns is the same in the `GROUP BY` clause and in the `ORDER BY` clause, the sort operation can be omitted for `ORDER BY` processing.

(c) Examples

There is an index that includes all GROUP BY columns as its component columns (for HiRDB/Single Server)

```
SELECT C1,C2,SUM(C3) FROM T1 GROUP BY C1,C2 ORDER BY C1,C2
               ↓
SELECT C1,C2,SUM(C3) FROM T1 GROUP BY C2,C1 ORDER BY C1,C2
```

There is no such index that includes all GROUP BY columns as its component columns

```
SELECT C1,C2,SUM(C3) FROM T1 GROUP BY C1,C2 ORDER BY C2,C1
               ↓
SELECT C1,C2,SUM(C3) FROM T1 GROUP BY C2,C1 ORDER BY C2,C1

SELECT C1,C2,SUM(C3) FROM T1 GROUP BY C1,C2 ORDER BY C2 DESC
               ↓
SELECT C1,C2,SUM(C3) FROM T1 GROUP BY C2,C1 ORDER BY C2 DESC
```

(d) Note

For the `GROUP BY` clause, define an index described in *17.6.2(4) Index definitions for a table used for retrieval specifying ORDER BY or GROUP BY*, if possible.

17.6.4 Notes about join retrieval

(1) Nested loop join using a partitioning key

(a) Checking the display

With a HiRDB/Parallel Server, if `Join Type` is `NESTED LOOPS JOIN` and the transfer information is `BROADCAST`, `KEY RANGE PARTIAL BROADCAST`, or `PARTIAL BROADCAST`, you may be able to improve performance by changing the partitioning key of an inner table for nested loop join.

This is not so effective when there are not many data items in the table, but it is effective when there are many partitions.

(b) Better method

If partitioning the inner table subject to nested loop join during join retrieval, you can achieve parallel processing by defining the table in such a manner that its partitioning key becomes the joined column. Note that parallel processing is not available with flexible hash partitioning. Make sure that the transfer information is *number*`-CLM HASH`, *number*`-CLM KEY RANGE`, or `1TO1`, or number`-CLM MULTIDIM`.

(c) Reason

If the inner table subject to nested loop join is partitioned by the key range or hash partitioning method, and the inner table's partitioning keys are all included in the joined columns, the inner table search range can be narrowed by the table partitioning conditions, thereby reducing the number of input/output operations as well as the CPU time.

(d) Example

```
SELECT * FROM T1, T2 WHERE T1.C1=T2.C1 and T1.C1='a'
           ↓
CREATE TABLE T1 (C1 INT NOT NULL, C2 INT)
    FIX HASH HASH6 BY C1 IN (USER1,USER2,USER3,USER4)
CREATE TABLE T2 (C1 INT NOT NULL, C2 INT)
    FIX HASH HASH6 BY C1 IN (USER1,USER2,USER3,USER4)
```

(e) Note

If the flexible hash partitioning method is used, the utility always assumes `BROADCAST` for the transfer information because it cannot identify the data storage location; therefore, parallel processing is not possible. If possible, you should use the FIX hash partitioning method instead of the flexible hash partitioning method. For details about these partitioning methods, see the *HiRDB Version 8 Installation and Design Guide*.

Chapter

18. Database Copy Utility (pdcopy)

This chapter explains how to use the database copy utility (`pdcopy`), which makes a backup of a database.

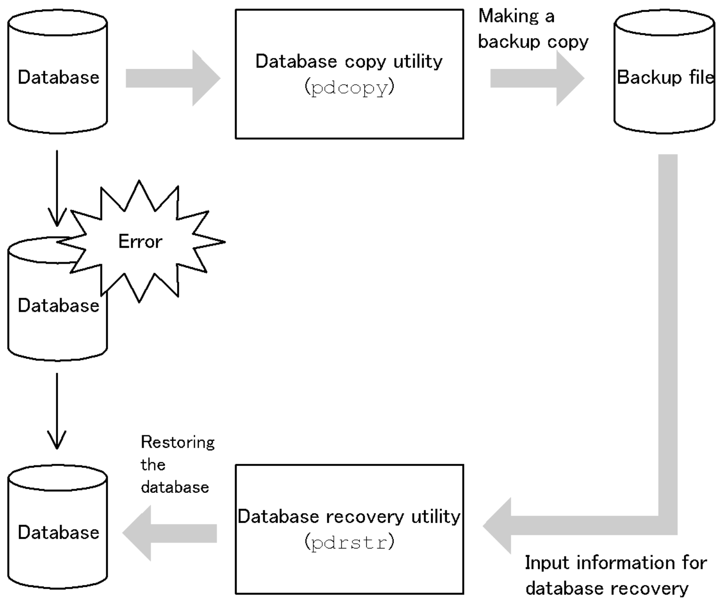
This chapter contains the following sections:

18.1 Functions of the database copy utility

Executor: HiRDB administrator

The HiRDB administrator must periodically back up databases to protect against possible database errors. To back up a database, use the database copy utility (the obtained backup can be used only to restore the same HiRDB's database). To restore the database due to a disk error, etc., use the created backup copy as an input to the database recovery utility (`pdrstr` command) to restore the database. Figure 18-1 shows an overview of database recovery.

Figure 18-1: Overview of database recovery



18.2 Information required for making backup copies

(1) Backup units

You can make backup copies in the following units. The database copy utility provides an option to specify these backup units. Table 18-1 shows the backup units.

Table 18-1: Backup units

Backup unit	Description	Option
By the system	Backs up all RDAREAs in the system (including master directory RDAREAs), except the user RDAREAs. The list RDAREAs are not subject to backup processing.	`-a`
By the unit*	Backs up all RDAREAs in a specified unit.	`-u`
By the server*	Backs up all RDAREAs in a specified server.	`-s`
By the RDAREA	Backs up specified RDAREAs. A group of RDAREAs can be backed up by specifying its RDAREA names as a regular expression.	`-r`

* This is applicable to a HiRDB/Parallel Server only, not applicable to a HiRDB/Single Server.

(2) Backup acquisition mode

The database copy utility's `-M` option lets you select a backup acquisition mode. Table 18-2 lists available backup acquisition modes.

Table 18-2: Backup acquisition modes

Backup acquisition mode (value of -M option)	Description	Database recovery method
`x`	This mode does not allow an RDAREA under backup processing to be referenced or updated. Use the `pdhold` command to place the RDAREA subject to backup processing in the shutdown and closed status.	Only the backup copy is needed to restore the database to the status existing when the backup copy was made.
`r`	This mode allows an RDAREA under backup processing to be referenced but does not allow it to be updated.	

Backup acquisition mode (value of -M option)	Description	Database recovery method
`s`	This mode allows an RDAREA to be referenced or updated while it is being backed up. If LVM (logical volume manager) is used for the database, the RDAREA to be backed up must be placed in backup hold status. This option is applicable when the RDAREA to be backed up is created in a character special file.	The backup copy and the system log obtained during backup processing* are needed to restore the database.

* The database copy utility process results listing contains the name and generation number of the system log file required for restoring the RDAREA. For details about the database copy utility process results listing, see Section *18.6 Database copy utility process results listing*.

Note

To back up all RDAREAs during HiRDB operation (in units of systems), specify `r` or `s` for the backup acquisition mode (`-M` option). You cannot specify `x` for the following reason: to specify `x`, you need to place all RDAREAs subject to backup processing in the shutdown and closed status using the `pdhold` command. However, the master directory RDAREA cannot be placed in the shutdown and closed status. Therefore, you cannot specify `x` in the `-M` option to back up all RDAREAs during HiRDB operation.

(3) Backup timing

You should back up data periodically as well as at the following times. If backups are not made at these times, the databases cannot be restored in the event of an error.

- Before and after executing a UAP or utility in the no-log mode, when applicable
- After executing a UAP or utility in the pre-update log acquisition mode, when applicable
- After adding, expanding, or re-initializing RDAREAs
- After recovering a database with the `pdstart -r` command
- After transferring dictionary information with the dictionary import/export utility
- After recovering a database from log information
- After upgrading HiRDB

(4) RDAREAs that need to be backed up at the same time

When backing up a specific RDAREA, you also need to back up those RDAREAs that are updated by the RDAREA backup processing (such as data dictionary RDAREAs). For details about the RDAREAs that need to be backed up at the same time, see the

HiRDB Version 8 System Operation Guide.

(5) Server machine storing backup files

You can create a backup file in any server machine where HiRDB is running. There is no need to create a backup file in the same server machine that contains the corresponding RDAREA. You might want to create backup files in the server machine that has a CMT device.

(6) Creating a backup file in a character special file

To create a backup file in a character special file, define that character special file as a HiRDB file system area for the utility. To do this, specify `UTL` in the `-k` option of the `pdfmkfs` command.

(7) Operation without unloading the system log

If you are running the system without unloading the system log, you need to obtain a log point information file when making a backup copy. To obtain a log point information file, specify the `-z` option with the database copy utility.

For details about how to operate without unloading the system log, see the *HiRDB Version 8 System Operation Guide*.

(8) Creating a backup using the differential backup facility

The differential backup facility backs up only the information that has changed since the previous backup, thereby reducing the backup processing time. Consider using the differential backup facility when the database is large and the amount of data that has been updated is small.

To use the differential backup facility, specify the `-g`, `-d`, `-K`, `-o`, and `-L` options.

For details about using the differential backup facility, see the *HiRDB Version 8 System Operation Guide*.

(9) Creating a backup using the inner replica facility

The inner replica facility is used when there is not enough time to make a backup of the database, such as when online applications are running 24 hours a day, 365 days a year. The inner replica facility permits the database to be referenced and updated while the database is being backed up.

To make a backup using the inner replica facility, specify the `-q` option.

For details about the inner replica facility and the required products, see the *HiRDB Staticizer Option Version 7 Description and User's Guide*.

(10) HiRDB status during database copy utility execution

1. You can execute the database copy utility only when HiRDB is active.
2. Execute the database copy utility at the server machine containing the single

server or the server machine where the system manager is located.

3. To execute the database copy utility, the unit containing the RDAREA subject to backup processing and the unit in which a backup file is to be created must be active. If you specify `x` or `r` in the `-M` option, the server containing the RDAREA subject to backup processing may not be active, but if you specify `s` in the `-M` option, it must be active.

(11) Size of a backup file

If the disk has enough space, but a message indicating a shortage of disk space is displayed during execution of the database copy utility, the size of the backup file may be greater than 2 gigabytes. HiRDB can only handle files whose size is less than 2 gigabytes. If this is the case, take one of the following actions:

(a) Action 1

Specify multiple backup files. For example, if you are backing up 3 gigabytes of data, specify at least two backup files.

(b) Action 2

Use a regular file on a UNIX system for which the option allowing use of large files (size of 2 gigabytes or more) was specified during the construction of the UNIX file system, or use a HiRDB file system area. For details about creating a large file in a HiRDB file system area, see the *HiRDB Version 8 Installation and Design Guide*.

(c) Action 3

Create the backup file in a HiRDB file system area in a character special file. For details about creating a large file in a HiRDB file system area, see the *HiRDB Version 8 Installation and Design Guide*.

18.3 Examples

This section presents examples of making backup copies. The topics include:

- Backing up in units of systems
- Backing up in units of RDAREAs
- Backing up in units of servers
- Backing up to files on DAT
- Backing up using JP1/OmniBack II
- Backing up using the differential backup facility
- Backing up using the inner replica facility and specifying an RDAREA generation
- Backing up using NetBackup

For details about the `pdcopy` command options used in these examples, see Section *18.4 Command format*.

18.3.1 Backing up in units of systems

This example backs up all RDAREAs:

```
pdcopy -m /hirdb/rdarea/mast/mast01 -M r -p /usr/hirdb/pdcopy/list/list01
-z /usr/hirdb/pdcopy/logpoint/logpoint01 -b /usr/hirdb/pdcopy/backup/backup01 -a
```

Explanation

`-m`: Specifies the name of the first HiRDB file in the master directory RDAREA.

`-M`: Specifies the backup acquisition mode.

`-p`: Specifies the output file name for the database copy utility process results listing.

`-z`: Specifies the name of the log point information file. There is no need to specify this option if the system operation involves unloading of the system log. This option is applicable to a HiRDB/Single Server but not to a HiRDB/Parallel Server.

`-b`: Specifies the name of the backup file.

`-a`: Specifies that all RDAREAs are to be backed up.

18.3.2 Backing up in units of RDAREAs

This example backs up two user RDAREAs (`RDAREA01` and `RDAREA02`):

```
pdcopy -m /hirdb/rdarea/mast/mast01 -M r -p /usr/hirdb/pdcopy/list/list01
-b /usr/hirdb/pdcopy/backup/backup01 -r RDAREA01,RDAREA02
```

Explanation

`-m`: Specifies the name of the first HiRDB file in the master directory RDAREA.

`-M`: Specifies the backup acquisition mode.

`-p`: Specifies the output file name for the database copy utility process results listing.

`-b`: Specifies the name of the backup file.

`-r`: Specifies the name of the RDAREAs to be backed up (`RDAREA01` and `RDAREA02`).

18.3.3 Backing up in units of servers

This example backs up all RDAREAs in a back-end server (`bes1`):

```
pdcopy -m /hirdb/rdarea/mast/mast01 -M r -p /usr/hirdb/pdcopy/list/list01
-z /usr/hirdb/pdcopy/logpoint/logpoint01 -b /usr/hirdb/pdcopy/backup/backup01
-s bes1
```

Explanation

`-m`: Specifies the name of the first HiRDB file in the master directory RDAREA.

`-M`: Specifies the backup acquisition mode.

`-p`: Specifies the output file name for the database copy utility process results listing.

`-z`: Specifies the name of the log point information file. There is no need to specify this option if the system operation involves unloading of the system log.

`-b`: Specifies the name of the backup file.

`-s`: Specifies that all RDAREAs in the back-end server (`bes1`) are to be backed up.

18.3.4 Backing up to files on DAT

This example backs up all RDAREAs to files on DAT:

```
pdcopy -m /hirdb/rdarea/mast/mast01 -M r -p /usr/hirdb/pdcopy/list/list01
-z /usr/hirdb/pdcopy/logpoint/logpoint01 -b /dev/rmt/0m -a
```

Explanation

`-m`: Specifies the name of the first HiRDB file in the master directory RDAREA.

`-M`: Specifies the backup acquisition mode.

`-p`: Specifies the output file name for the database copy utility process results listing.

`-z`: Specifies the name of the log point information file. There is no need to specify this option if the system operation involves unloading of the system log. This option is applicable to a HiRDB/Single Server but not to a HiRDB/Parallel Server.

`-b`: Specifies the name of the backup file.

`-a`: Specifies that all RDAREAs are to be backed up.

18.3.5 Backing up using JP1/OmniBack II

This example uses JP1/OmniBack II to back up all RDAREAs in batch mode:

```
pdcopy -m /hirdb/rdarea/mast/mast01 -M r -p /usr/hirdb/pdcopy/list/list01
-z /usr/hirdb/pdcopy/logpoint/logpoint01 -b host01:backup001 -a -k o -G DLT01
```

Explanation

`-m`: Specifies the name of the first HiRDB file in the master directory RDAREA.

`-M`: Specifies the backup acquisition mode.

`-p`: Specifies the output file name for the database copy utility process results listing.

`-z`: Specifies the name of the log point information file. There is no need to specify this option if the system operation involves unloading of the system log. This option is applicable to a HiRDB/Single Server but not to a HiRDB/Parallel Server.

`-b`: Specifies the name of the backup file.

`-a`: Specifies that all RDAREAs are to be backed up.

`-k`: Specifies the type of backup file. This example specifies o to back up RDAREAs to JP1/OmniBack II's object.

-G: Specifies the name of the barlist file. The following shows the contents of the barlist file (DLT01):

```
BARLIST "DLT01"   ..........1
DEFAULTS
{
}
DEVICE "backup01"   ........2
{
}
CLIENT HiRDB host4   .......3
{
  -public   ................4
} -protect none   ..........5
```

Explanation

1. Name of the barlist file
2. Name of the logical device (backup file)
3. Name of the host containing JP1/OmniBack II's disk agent
4. Required operand
5. Object protection specification (not protected)

18.3.6 Backing up using the inner replica facility and specifying an RDAREA generation

This example backs up a specified generation of RDAREAs using the inner replica facility.

```
pdcopy -m /hirdb/rdarea/mast/mast01 -b /bkdir/bkup02
-r MAST,DIR,DIC,USR01,USR02,USR03 -q 1
```

Explanation

-m: Specifies the name of the first HiRDB file in the master directory RDAREA.

-b: Specifies the name of the backup file.

-r: Specifies the names of the RDAREAs to be backed up (MAST, DIR, DIC, USR01, USR02, and USR03).

-q: Specifies the generation number of the RDAREAs to be backed up.

18.3.7 Backing up using the differential backup facility

This example backs up user RDAREAs (RDAREA01 and RDAREA02) using the differential backup facility. The example assumes that a full backup has already been made with `-d a` specified.

```
pdcopy -m /rdarea/mast/mast01 -M r -g backupg1 -b /pdcopy/backup02
-d -K /pdcopy/admfile -o /pdcopy/rfile
```

Explanation

`-m`: Specifies the name of the first HiRDB file in the master directory RDAREA.

`-M`: Specifies the reference-possible mode as the backup acquisition mode.

`-g`: Specifies the name of the differential backup group. Specification of `(S)` is not required.

`-b`: Specifies the name of the backup file (differential backup file name).

`-d`: Specifies the backup type (this example specifies `-d` to make a differential backup).

`-K`: Specifies the name of the HiRDB file system area used to store the differential backup management file.

`-o`: Specifies the name of the historical file for differential backups.

18.3.8 Backing up using NetBackup

This example makes a backup of all RDAREAs in the batch mode using NetBackup.

```
pdcopy -m /hirdb/rdarea/mast/mast01 -M r -p /usr/hirdb/pdcopy/list/list01
-z /usr/hirdb/pdcopy/logpoint/logpoint01 -b host01:POLICY01 -a -k n
```

Explanation

`-m`: Specifies the name of the first HiRDB file in the master directory RDAREA.

`-M`: Specifies the backup acquisition mode.

`-p`: Specifies the output file name for the database copy utility processing results listing.

`-z`: Specifies the name of the log point information file. There is no need to specify this option if the system operation involves unloading of the system log. This option is applicable to a HiRDB/Single Server but not to a HiRDB/Parallel Server.

`-b`: Specifies the name of the NetBackup policy as the backup file name, in the format *post-name*`:`*policy-name*.

`-a`: Specifies that all RDAREAs are to be backed up.

-k: Specifies the type of backup file. This example specifies n to make a backup using NetBackup.

18.4 Command format

18.4.1 Option format

The required or frequently specified options are indicated in bold. In the table below, each number corresponds to the number assigned to each option, which is explained in *18.4.2 Options*.

When separating an option's flag arguments with the comma, do not specify spaces before or after the comma. If a space is specified, the command ignores whatever is specified following the space.

No.	Option
1	**pdcopy -m [*host-name*:]*first-HiRDB-file-name-in-master-directory-RDAREA***
2	**[-M {<u>x</u>|r|s}]**
3	**[-p *process-results-output-file-name*]**
4	[-i]
5	[-j *lock-release-wait-time*]
6	[-E *EasyMT-MT-attributes-definition-file-name*]
7	[-B *EasyMT-I/O-buffer-sectors-count*]
8	[-z *log-point-information-file-name*]
9	[-J]
10	[-w *pause-time*,*read-count*]
11	[-f *control-statement-file-name**]

* The options that can be specified in the control information file are shown below. You can also specify these options directly in the pdcopy command.

Contents of control statement file

No.	Option
12	**-b {[*host-name*:]*backup-file-name*[,*backup-file-name*]...** **|[*host-name*:]*device-symbolic-name*[,*device-symbolic-name*]** **|[*host-name*:]*device-group-name*** **|[*host-name*:]*object-name*** **|[*host-name*:]*policy-name*}**
13	**{ -a**

No.	Option
14	**`|-u` *unit-identifier*`[,`*unit-identifier*`]...`**
15	**`|-s` *server-name*`[,`*server-name*`]...`**
16	**`|-r` *RDAREA-name*`[,`*RDAREA-name*`]... }`**
17	**`[-k {u|i|e|m|o|n}]`**
18	`[-v` *volume-name*`[,`*volume-name*`]...]`
19	`[-N` *EasyMT-file-name*`]`
20	`[-G` *barlist-file-name*`]`
21	`[-g` *differential-backup-group-name*`]`
22	`[-d` *backup-type*`]`
23	`[-K` *HiRDB-file-system-area-name-storing-differential-backup-management-file*`]`
24	`[-o` *differential-backups-history-file-name*`]`
25	`[-L` *differential-backups-management-file-size*`]`
26	`[-q` *generation-number*`]`

27	`[-S{`*backup-file-initial-size*`,`*extension-value*
	`|'(`*backup-file-initial-size*`,`*extension-value*`)`
	`[,(`*backup-file-initial-size*`,`*extension-value*`)]...'}]`

- Specifiability of options when another backup program is linked

Option	Other backup program		
	EasyMT	JP1/OmniBack II	NetBackup
`-m`	M	M	M
`-M {x|r|s}`	O	O	O
`-p`	O	O	O
`-i`	O	O	O
`-j`	O	O	O
`-E`	O	N	N
`-B`	O	N	N
`-z`	O	O	O

Option		Other backup program		
		EasyMT	JP1/OmniBack II	NetBackup
`-J`		O	O	O
`-w`		O	O	O
`-f`		O	O	O
`-b`		M	M	M
`{-a|-u|-s|-r}`		M[1]	M[1]	M[1]
`-k`	`u`	N	N	N
	`i`	N	N	N
	`e`	M[2]	N	N
	`m`	M[2]	N	N
	`o`	N	M	N
	`n`	N	N	M
`-v`		O	N	N
`-N`		O	N	N
`-G`		N	M	N
`-g`		O	O	O
`-d {a|b|c|d}`		O	O	O
`-K`		O	O	O
`-o`		O	O	O
`-L`		O	O	O
`-q`		O	O	O
`-S`		N	N	N

Legend:

M: Mandatory (must be specified)

O: Optional (can be specified)

N: Not permitted (cannot be specified)

1 You must specify one of `-a`, `-u`, `-s`, or `-r`.

2 You must specify either `e` or `m`.

18.4.2 Options

(1) -m [host-name:]first-HiRDB-filename-in-master-directory-RDAREA

~ <identifier:pathname> ((up to 167 characters))

Specifies the name of the first HiRDB file in the master directory RDAREA. Specify the name of the host containing the master directory RDAREA and the path name, separated by a colon (:). This is the name specified in the `pd_master_file_name` operand in the system common definitions.

Example

```
-m host01:/hirdb_s/rdarea/rdmast
```

Rules

1. You can omit the host name if the master directory RDAREA is located at the server machine where the database copy utility is executed (where the `pdcopy` command is entered). In this case, specify only the path name.
2. If you specify the host name, make sure that it is a host name specified in the `-x` option of the `pdunit` operand in the system definition.
3. If you are using the system switchover facility, make sure that the host name of the primary system is specified.

(2) -M{x|r|s}

Specifies the backup acquisition mode. The following table shows available backup acquisition modes:

-M Option	Description	Database recovery method
`x` (Reference/update-impossible mode)	This mode does not allow an RDAREA under backup processing to be referenced or updated. Use the `pdhold` command to place the RDAREA subject to backup processing in the shutdown and closed status.	Only the backup copy is needed to restore the database to the status existing when the backup copy was made.
`r` (Reference-possible mode)	This mode allows an RDAREA under backup processing to be referenced but does not allow it to be updated.	

-M Option	Description	Database recovery method
s (Updatable mode)	This mode allows an RDAREA under backup processing to be referenced or updated. To use this mode, the RDAREA subject to backup processing must be created in a character special file.	To restore the database, you need the backup copy and the system log obtained at the synchronization point immediately before the backup*.

Notes

1. To back up the master directory RDAREA with `-M x` specified, start HiRDB using the `pdstart -r` command, then back up the master directory RDAREA using the database copy utility.

   If you specify `-M r`, you can back up the master directory RDAREA with the `pdcopy` command by starting HiRDB with the normal `pdstart` command.

2. If `RECOVERY PARTIAL` or `RECOVERY NO` is specified in `CREATE TABLE` for a LOB column (including when the `RECOVERY` operand is omitted), `-M s` does not back up the user LOB RDAREA containing the LOB column nor the data dictionary LOB RDAREA storing the object.

3. If `-M s` is specified and another transaction is updating a page subject to backup processing, the utility waits until the transaction issues a `COMMIT`. Therefore, `pdcopy` may result in a time-out.

4. If you are executing `pdcopy` in the updatable mode using an LVM for the database, you must place the target RDAREAs in backup hold status.

* The database copy utility process results listing contains the name and generation number of the system log file required for restoring the RDAREA. For details about the database copy utility process results listing, see Section *18.6 Database copy utility process results listing*.

Notes

1. To back up all RDAREAs during HiRDB operation (in units of systems), specify `r` or `s` as the backup acquisition mode (`-M` option). You cannot specify `x` for the following reason: to specify `x`, you need to place all RDAREAs subject to backup processing in the shutdown and closed status using the `pdhold` command. However, the master directory RDAREA cannot be placed in the shutdown and closed status. Therefore, you cannot specify `x` in the `-M` option to back up all RDAREAs during HiRDB operation.

2. You cannot make a backup copy specifying `-M s` while a UAP or utility is executing in the no-log mode or pre-update log acquisition mode. To back up data after completion of a UAP or utility in the no-log mode or pre-update log

acquisition mode, you need to specify `-M r` or `-M x`.

3. When you make a backup copy with `-M s` specified in either of the following cases, you must first use the `pdlogswap -w` command to swap system logs:
   - When `pdmod` has been executed (to exclude configuration modification processing from the system log file)
   - After operating in the no-log mode (to exclude no-log processing from the system log file)
4. For a shared RDAREA, `-M s` cannot be specified to acquire a backup.

(3) -p process-results-output-filename

~ <pathname>

Specifies the name of the file to which the processing result of the database copy utility is to be output.

Example

```
-p /usr/pdcopy/list01
```

Rules

1. Specify the path name of the server machine at which the database copy utility is executed (where the `pdcopy` command is entered).
2. If the specified path name is not found or this option is omitted, the database copy utility outputs its processing result to the server machine where the utility is executed. The utility displays the output destination in the `KFPR26022-I` message.
3. Regardless of the specification of this option, the utility outputs all error messages to the system log file and standard output, and the final processing result to the standard output. However, error messages may not be output to the system log file and to the standard output in the same order.

(4) -i

Specifies that utilization status is to be output to the process results output file. This option is applicable to the following RDAREAs:

- Data dictionary RDAREAs
- Data dictionary LOB RDAREAs
- User RDAREAs
- User LOB RDAREAs

The following information is output to the process results output file:

Type	Informatio n	Description
Page	Page count	• Number of available pages • Number of unused pages • Number of used pages ● (a) Number of pages storing index[1] ● (b) Number of pages storing table[1]
	Ratio	• Ratio of unused pages to available pages • Ratio of used pages to available pages • Ratio of pages storing index to used pages[1] • Ratio of pages storing table to used pages[1]
Area length[1]	Length	• Sum of free areas in used pages • Sum of free areas in pages storing index • Sum of free areas in pages storing table • Area length of all pages storing index • Area length of all pages storing table
	Ratio	• Ratio of sum of free areas to the area length of all pages storing index • Ratio of sum of free areas to the area length of all pages storing table
System information fragmentation rate[2]	Ratio	Extent of fragmentation of system information in data dictionary LOB RDAREAs and user LOB RDAREAs[3]

[1] This information is not output for data dictionary LOB RDAREAs or user LOB RDAREAs.

[2] This information is output only for data dictionary LOB RDAREAs and user LOB RDAREAs.

[3] A value greater than 1% can have adverse effects on access performance, in which case you need to reorganize the database using the database reorganization utility. For a data dictionary LOB RDAREA, reorganize the dictionary tables related to stored procedures (`SQL_ROUTINES`, `SQL_ROUTINE_RESOURCES`, `SQL_ROUTINE_PARAMS`).

(5) -j lock-release-wait-time

~ <unsigned integer> ((0-200,000))

Specifies in seconds the maximum wait time for monitoring lock release.

Lock-release wait time is the time from when a lock request enters a wait status, until the status is released. If the lock is not released within the specified time, `pdcopy` terminates with an error.

If you specify a value of `0`, the utility waits until the lock is released, without monitoring the lock-release wait time. If you omit this operand, the utility assumes the value of the `pd_lck_wait_timeout` operand in the system definitions.

Notes

1. If `-M r` is specified and the specified RDAREA is already locked by another user, `pdcopy` terminates with an error.
2. If `-M s` is specified and a requested page is being released by another user, `pdcopy` is placed in lock-release wait status until the user's transaction is terminated. In this case, specify a lock-release wait time that is greater than the execution time of the transaction conducting the page release. Page release occurs mainly in the following cases:
    - `DROP TABLE`
    - `DROP INDEX`
    - `PURGE TABLE` statement
    - `INSERT` or `UPDATE` statement executed on a column with an index
    - `DELETE` statement executed on a column with an index (with duplicate key)
    - `DELETE` statement executed after `LOCK` statement; or, `UPDATE` statement executed after `LOCK` statement resulting in a change to row length

(6) -E EasyMT-MT-attributes-definition-filename

~ <path name>

Specifies the name of the EasyMT MT attributes definition file. This file must be connected to the server machine where the database copy utility is executed (where the `pdcopy` command is entered). The `-E` option is applicable to a backup file that has been created with e or m specified in the `-k` option. The following attributes take effect:

- buffno: Number of I/O buffer sectors
- magazin: MT device allocation pattern
- job: Job name
- expire: Expiration date
- preserve: Number of days to be saved

When the `-B` option is specified, it takes precedence over the number of I/O buffer sectors specified in the MT attributes definition file.

The utility checks the information specified in this file when EasyMT is executed.

(7) -B EasyMT I/O buffer sectors count

~ <unsigned integer> ((1 255))

Specifies the number of I/O buffer sectors to be used for magnetic tape input/output operations. A larger value results in better performance, but it requires more memory.

Rules

1. The `-B` option is applicable to a backup file that has been created with e or m specified in the `-k` option.
2. If the `-E` and `-B` options are both omitted, the utility assumes EasyMT's default values.

(8) -z log-point-information-file-name

~ <pathname>

This option is applicable to an operation that does not involve unloading of the system log. Specify the name of the file used to store the log point information.

Example

```
-z /usr/pdcopy/logp01
```

Rules

1. Specify the path name of the server machine where the database copy utility is executed (where the `pdcopy` command is entered).
2. If this log point information file is specified in the `pdlogchg` command, any unneeded system log file created before the log point is placed in the unload completed status. When this option is specified, the utility also stores the log point information in the backup file.

Notes

1. For a HiRDB/Parallel Server, for RDAREAs subject to backup processing, specify only the names of those RDAREAs that are located at the same server.
2. The server containing the RDAREAs subject to backup processing may be active when the utility is executed.
3. If a backup copy is to be made by starting HiRDB with the `pdstart -r` command, the `-z` option cannot be specified. To back up all data by specifying the `-z` option, start HiRDB with `pdstart`, then specify `pdcopy -M r`. The utility obtains a log point information file for each server; therefore, for a HiRDB/Parallel Server, it is impossible to back up all data at the same time. Additionally, the master directory RDAREAs cannot be placed in the shutdown or closed status. Therefore, if the processing is started with `pdstart`, `pdcopy -M x -a` cannot be specified.

(9) -J

Specifies that when a skip target error is detected while copying RDAREAs, the next RDAREA is to be copied immediately without terminating the `pdcopy` command.

The `-J` option is mutually exclusive with the `-g`, `-K`, and `-d` options. If they are specified, specification of the `-J` option is ignored.

The table below describes the skip target errors. Note that a skip may not have occurred if the `KFPR26061-W` message was not issued before or after the error message.

Skip target error	Message issued for a skip target error
A name specified in the `-r` option is not defined as an RDAREA, or a list RDAREA was specified.	`KFPR26006-E`
`-M x` was specified, but the RDAREAs to be copied include the master directory RDAREA.	`KFPR26111-E`
An open or close error was detected before copy processing started.	`KFPR26012-E`
`-M s` was specified, but the server containing an RDAREA to be copied is not active.	`KFPR26015-E`
`recovery partial` or `recovery no` was specified during table definition and `-M s` was specified, but the RDAREAs to be copied include a user LOB RDAREA.	`KFPR26030-E`
`-M x` was specified, but an RDAREA to be copied is not in shutdown and closed status.	`KFPR26110-E`

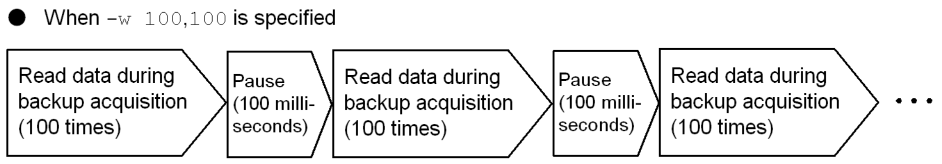
(10) -w pause-period,read-count

Because execution of `pdcopy` involves transfer of a large amount of data, the CPU usage rate increases and other online jobs may be affected adversely. Specifying the `-w` option pauses the utility for a specified period of time after reading a specific amount of data, thereby minimizing adverse effects on other online jobs.

Although specifying this option reduces the CPU usage rate, the `pdcopy` processing time increases.

Figure 18-2 provides an overview of the processing when the `-w` option is specified.

Figure 18-2: Overview of processing when the -w option is specified


● When -w 100,100 is specified
Read data during backup acquisition (100 times)
Pause (100 milli-seconds)
Read data during backup acquisition (100 times)
Pause (100 milli-seconds)
Read data during backup acquisition (100 times)
• • •

pause-period ~ <unsigned integer> ((10 ~ 60000))

Specifies the pause period during data read operations in units of 10 milliseconds (example: 10, 20, 30,...). If the specified value is not a multiple of 10 milliseconds, it is rounded up.

The utility performs as many read operations as specified in *read-count* and then pauses for the specified period.

read-count ~ <unsigned integer> ((1 ~ 10000))

Specifies the number of read operations to be performed before the processing is paused.

The utility performs as many read operations as specified here and then pauses for the specified period.

Rules

1. In determining the pause period and read count, take into account the total size of the data to be read by `pdcopy` and the total number of read operations.
2. For a HiRDB/Parallel Server, the processing specified with the `-w` option is performed for each server. Read operations are counted for each server and pause processing is performed for each server. Check the server that contains the target RDAREA before specifying the pause period and read count.

Determining the processing time when the -w option is specified

1. Obtain the data read count for each RDAREA.
   - For a data directory RDAREA

     $\uparrow$(4096 $\times$ number of records in the HiRDB file system)/65536$\uparrow$

     + (number of HiRDB files constituting the RDAREA - 1)
   - For other RDAREA

     $\uparrow$Page length $\times$ (number of records in the HiRDB file system

     - (number of unused segments $\times$ segment size))/65536$\uparrow$

     + (number of HiRDB files constituting the RDAREA - 1)
2. Obtain the total number of data read operations.
   - For a HiRDB/Single Server

     Sum of the numbers of data read operations for all target RDAREAs
   - For a HiRDB/Parallel Server

     Largest sum of the numbers of data read operations for all target RDAREAs among all applicable servers
3. Obtain the processing time.

↓(total number of data read operations obtained in 2/number of read operations)↓

× pause period (milliseconds)

+ `pdcopy` processing time

(11) -f control-information-filename

～ <pathname>

Specifies the name of the control information file.

Example

```
-f /usr/pdcopy/cont01
```

This control statement must contain the options beginning at (12) as follows. You can also specify these options directly in the `pdcopy` command.

Notes

1. Create the control statement file at the server where the database copy utility is executed (where the `pdcopy` command is entered).
2. Specify in the control statement file at least one set of a backup file name, options for the RDAREAs subject to backup processing, a file type, a volume name, and an EasyMT file name. Note that each set of this information must be specified on a single line (which is up to 32,768 bytes). You can specify a maximum of 16 lines of information.
3. If you are using the differential backup facility with the `-a` or `-q` option specified, you can specify only one control statement line.

Format of the control statement file

```
 -b flag-argument {-a|-u flag-argument|-s flag-argument|-r flag-argument}
                   [-k flag-argument][-v flag-argument][-N flag-argument]
                   [-S flag-argument][-G flag-argument]
[-b flag-argument [{-u flag-argument|-s flag-argument|-r flag-argument}]
                   [-k flag-argument][-v flag-argument][-N flag-argument]
                   [-S flag-argument]][-G flag-argument]
[-b flag-argument [{-u flag-argument|-s flag-argument|-r flag-argument}]
                   [-k flag-argument][-v flag-argument][-N flag-argument]
                             [-S flag-argument]][-G flag-argument]
                         .
                         .
                         .
```

Notes

1. For details about each option flag, see the explanation of the `-b` option as

follows.

2. When specifying multiple lines, you can specify only those option flags specified on the first line also on the subsequent lines (except for the `-a` option). For example, if you specify the `-u` option on the first line, you can specify the `-u` option on any subsequent line.

(12) -b {[host-name:]backup-file-name[,backup-file-name]...|[host-name:]device-symbolic-name[,device-symbolic-name]|[host-name:]device-group-name|[host-name:]object-name|[host-name:]policy-name}

~ <identifier: path name> ((up to 167 characters when `-k i` is specified))

Specifies the backup file name, object name of JP1/OmniBack II, or policy name specified with NetBackup.

- When specifying a backup file name, device symbolic name, or device group name

  Specify the name of the host that will contain the backup files, separated by a colon (`:`) from the names of the backup files, or from the device symbolic names, or from the device group name. Specify a path name for a backup file name, and specify an identifier for a device symbolic name or device group name.

  ```
  Example: -b host01:/usr/pdcopy/backup01
  ```

- When specifying an object name of JP1/OmniBack II

  When using JP1/OmniBack II, specify the name of the host where a disk agent of JP1/OmniBack II is located, separated by a colon (`:`) from an object name. Note that an object name containing an underscore (_) or a hyphen cannot be specified.

  ```
  Example: -b host01:backup001
  ```

- When specifying a policy name of NetBackup

  When using NetBackup, specifying the name of the host that contains the NetBackup client, separated by a colon (:) from a policy name. Note that a policy name containing a hyphen cannot be specified.

  ```
  Example: -b host01:POLICY01
  ```

Rules

1. If you are creating a backup file at the server machine where the database copy utility is executed (where the `pdcopy` command is entered), you can omit the host

name, in which case you need to specify the path name only.

2. If you are specifying a host name, make sure that it is a host name specified in the `-x` option of the `pdunit` operand in the system definition.
3. If the server machine used to execute the database copy utility (where the `pdcopy` command is entered) has a disk agent of JP1/OmniBack II or a NetBackup client, you can omit the host name.
4. If you are using a control statement file, make sure that there is no duplication of backup file names on a line.
5. If you execute backup processing with an existing backup file name specified, that backup file will be overwritten.
6. A maximum of 1,120 RDAREAs can be backed up into a single backup file.
7. To back up RDAREAs into a file under a host using the system switchover facility, make sure that the host name of the primary system is specified. The utility creates a backup file under the host of the running system.
8. When using a disk agent of JP1/OmniBack II or a NetBackup client at a host that uses the system switchover facility, make sure that the host name of the primary system is specified. The disk agent of JP1/OmniBack II or NetBackup client will be the running system.
9. If you are using NetBackup and making a backup copy using the same policy name, do not change the RDAREAs to be backed up. If the policy name is the same but some of the RDAREAs are different, the system may not be able to restore the RDAREAs.
10. The following table describes the relationship between the `-b` and `-k` options:

-k option specification	-b option specification
`-k u`	You can specify one or more regular file names or one MT special file name. The second and subsequent files will be used only when there is more backup data than can be accommodated in the first file. Therefore, you need to specify in such a manner that each file is allocated to a different partition.
`-k i`	You can specify multiple sets of `"`*HiRDB-file-system-area-name*`/` *HiRDB-file-name*`"`. The second and subsequent files will be used only when there is backup data that can be accommodated in the first file. You must create the HiRDB file system area as a utility-dedicated area in advance with the `pdfmkfs` command.
`-k e`	You can specify an MT special file name.
`-k m`	You can specify a maximum of two device symbolic names or one device group name managed by MTguide. The system does not check whether a device symbolic name or a device group name is specified.

-k option specification	-b option specification
`-k o`	You can specify any name with a maximum length of 512 bytes. The specified name becomes the object name of JP1/OmniBack II. For the host name, specify the name of the host that has both a disk agent of JP1/OmniBack II and HiRDB. You can check the object names with the `omnidb` command (with the `-stream` option specified) of JP1/OmniBack II.
`-k n`	You can specify the policy name specified with NetBackup. Express a policy name as 1-128 bytes of alphanumeric characters. For the host name, specify the name of a host that has both NetBackup client and HiRDB. You can check the policy names with the `bpimagelist` command of NetBackup.

Notes

1. When you create multiple backup files, note the following:
   - NFS files cannot be used.
   - A tape device cannot be used directly without EasyMT.
   - When you specify multiple backup files, use regular files or HiRDB files.
2. If you specify the same policy name to execute `pdcopy` more than once, you can specify a backup file using the `-U` option's time specification when the database is restored (during execution of `pdrstr`). For the time of the backup file, specify the date and time of the `KFPR26071-I` message that is displayed during execution of `pdcopy`.

(13) -a

Specifies that RDAREAs are to be backed up in units of systems (all RDAREAs in the system are to be backed up).

Rules

1. If you specify the `-a` option, the control statement file can contain only one line of control statement.
2. If you specify the `-a` option when a replica RDAREA has been defined, the replica RDAREA will also be backed up.
3. If an open error occurs on a HiRDB file in a replica RDAREA, the utility skips backup of that RDAREA and backs up another RDAREA. If the differential backup facility is used and an open error occurs in a HiRDB file in a replica RDAREA, the utility terminates with an error.
4. If an input/output error occurs while a HiRDB file in a replica RDAREA is being read, the utility terminates with an error.

(14) -u unit-identifier[,unit-identifier]. . .

~ <identifier> ((4 characters))

Specifies unit identifiers when RDAREAs are to be backed up in units of units (all RDAREAs in the specified units are to be backed up).

Rules

1. If specifying multiple unit identifiers, make sure none of them is duplicated.
2. When using a control statement file, make sure that each specified unit identifier is unique among all the control statements.
3. If you specify the `-u` option when a replica RDAREA has been defined, the replica RDAREA will also be backed up.
4. If an open error occurs in a HiRDB file in a replica RDAREA, the utility skips backup of that RDAREA and backs up another RDAREA. If the differential backup facility is used and an open error occurs in a HiRDB file in a replica RDAREA, the utility terminates with an error.
5. If an input/output error occurs while a HiRDB file in a replica RDAREA is being read, the utility terminates with an error.

(15) -s server-name[,server-name]. . .

~ <identifier> ((1-8 characters))

Specifies the names of servers when RDAREAs are to be backed up in units of servers (all RDAREAs in the specified servers are to be backed up).

Rules

1. If specifying multiple server names, make sure none of them is duplicated.
2. When using a control statement file, make sure that each specified server name is unique among all the control statements.
3. If you specify the `-s` option when a replica RDAREA has been defined, the replica RDAREA will also be backed up.
4. If an open error occurs in a HiRDB file in a replica RDAREA, the utility skips backup of that RDAREA and backs up another RDAREA. If the differential backup facility is used and an open error occurs in a HiRDB file in a replica RDAREA, the utility terminates with an error.
5. If an input/output error occurs while a HiRDB file in a replica RDAREA is being read, the utility terminates with an error.

(16) -r RDAREA-name[,RDAREA-name]. . .

~ <identifier> ((1-30 characters))

Specifies the names of the RDAREAs to be backed up.

Rules

1. If specifying multiple RDAREA names, make sure none of them is duplicated.
2. When using a control statement file, make sure that each specified RDAREA name is unique among all the control statements (except for a regular expression).
3. You can specify RDAREA names in three different regular expressions, as follows. They enable a group of RDAREAs to be backed up collectively. When using a regular expression, be sure to use only one type of expression.

Regular expression	Explanation	Example
*character-string**	Any RDAREA name beginning with the specified character string	`PDUSER*` All RDAREAs with names beginning with `PDUSER` are processed.
character-string	Any RDAREA name containing the specified character string	`*PDUSER*` All RDAREAs with names containing the character string `PDUSER` are processed.
-	Any RDAREA name containing the specified character string at the specified location	`..USER` All RDAREAs with names containing the character string `USER` beginning in character 3 are processed.

Note

You can combine a regular expression of RDAREA names with individual RDAREA names.

Example

```
-r RDUSER*,RDMASTER
```

1. The following shows the rules for specifying RDAREA names:

Type	Coding method without using regular expression	Coding method using regular expression
Command line specification (without using a control statement file)	When specifying an RDAREA name in lowercase alphabetic characters or when the RDAREA name contains a space, enclose the entire name in double quotation marks (`"`). If you use the Bourne shell (sh), C shell (csh), or Korn shell (ksh), you also need to enclose the RDAREA name in single quotation marks (`'`).	Enclose the entire regular expression with single quotation marks (`'`). Regular expression is case sensitive.
Specifying in a control statement file	When specifying an RDAREA name in lowercase alphabetic characters or when the RDAREA name contains a space, enclose the entire name in double quotation marks (`"`).	Regular expression is case sensitive.

2. If the `-q` and `-r` options are both specified, only system RDAREAs and original RDAREAs can be specified. In this case, in the inner replica group to which an original RDAREA belongs, the replica RDAREA with the generation specified in the `-q` option is subject to processing.
3. If the replica RDAREA with the generation specified in the `-q` option is undefined, the utility ignores this RDAREA and backs up another RDAREA.
4. If an open error occurs in a HiRDB file in a replica RDAREA, the utility skips backup of that RDAREA and backs up another RDAREA. If the differential backup facility is used and an open error occurs in a HiRDB file in a replica RDAREA, the utility terminates with an error.
5. If an input/output error occurs while a HiRDB file in a replica RDAREA is being read, the utility terminates with an error.
6. If the `-q` option is specified, the regular expression is applicable to the original RDAREA. If the `-q` option is omitted, the regular expression is applicable to all RDAREAs including the replica RDAREAs.

(17) -k {u|i|e|m|o|n}

Specifies the type of backup file.

`u`

Specifies that the backup is to be made to a regular file or to MT without using EasyMT, JP1/OmniBack II, or NetBackup. You cannot specify this option if the backup file consists of multiple volumes (specify `-k m` instead).

`i`

Specifies that the backup is to be made to a HiRDB file system area. This option is applicable when the backup file is shared by the running system and a standby system, and the standby system is to be recovered using the backup made by the running system.

`e`

Specifies that the backup is to be made to magnetic tape using EasyMT. You cannot specify this option if the backup file consists of multiple volumes. This specification requires the use of EasyMT.

`m`

Specifies that the backup is to be made to magnetic tape using EasyMT and MTguide. You must specify this option if the backup file consists of multiple volumes. This specification requires the use of EasyMT and MTguide.

`o`

Specifies that the backup is to be made using JP1/OmniBack II. Specify in the`-b` option the object name of JP1/OmniBack II.

`n`

Specifies that the backup is to be made using NetBackup. Specify in the `-b` option the policy name set by NetBackup.

(18) -v volume-name[,volume-name]...

~ <alphanumerics> ((1-6 characters))

Specifies the volume names of the magnetic tapes to which the backup will be made. When this option is specified, an error results if a volume mounted at a tape deck does not match any of the specified values. If the number of volumes actually needed to make the backup is larger than the number of volumes specified here, the utility does not check any excess volume names. When this specification is omitted, the utility does not perform volume name checking.

This option is applicable to the backup file for which e or m is specified in the `-k` option. To specify multiple volume names, you need to specify m in the `-k` option.

Make sure that each specified volume name is unique among all volumes used.

(19) -N EasyMT-filename

~ <alphanumerics> ((1-17 characters))

Specifies the file name to be assigned when the backup file is created. This specification takes effect only on the backup file made with e or m specified in the `-k` option. The utility always creates a backup file as the first file on the mounted magnetic tape (beginning at file number 1).

(20) -G barlist-filename

~ <alphanumerics (excluding an underline (_))> <<1-64 characters>>

Specifies a barlist filename for JP1/OmniBack II.

A barlist file must be created at the server machine where JP1/OmniBack II's cell server is located.

Create the barlist file in the following directory:

`/etc/opt/omni/barlists/stream`

The following shows an example of a barlist file format:

Example

```
BARLIST "barlist-filename"   ..................1
DEFAULTS
{
}
DEVICE "logical-device-name"   ................2
{
}
CLIENT HiRDB host-name   ......................3
{
  -public  ..................................4
} -protect protection-specification   .........5
```

Explanation

1. Barlist file name (e.g., `BARLIST "DLT01FILE"`)
2. Logical device name (e.g., `DEVICE"DLT01"`)

   For details about the logical device, see the applicable JP1/OmniBack II manual.
3. Name of the host where the disk agent is located (e.g., CLIENT HiRDB h9000vr3)
4. Required operand
5. Protection specification

   `protection-specification={none|days n|weeks n|until Date|permanent}`

   `none`: Not protected.

   `days n`: For n, specify the number of days to be protected.

   `weeks n`: For n, specify the number of weeks to be protected.

`until Date`: For Date, specify the date at which protection expires in the format YYYY/MM/DD, enclosed in double quotation marks (`"`).

`permanent`: Permanent protection.

(21) -g differential-backup-group-name

~ <alphanumerics> ((1-30))

Specifies the name of a group as a unit subject to the differential backup facility.

If the specified differential backup group name ends with `S`, the utility immediately starts differential backup management, at which time the differential backup management file specified in the `-k` option is created. If an existing differential backup management file has the specified time, the utility deletes it. Therefore, you need to delete the backup files in the previous differential backup group.

For details about the differential backup facility, see the *HiRDB Version 8 System Operation Guide*.

Rules

1. The utility always obtains a full backup for user LOB RDAREAs, data dictionary LOB RDAREAs, and registry LOB RDAREAs.
2. For the same differential backup group, you cannot change the RDAREAs subject to backup processing.
3. Store all backup files belonging to the same differential backup group at the same host (specify only one host name in the `-b` option).
4. Create a differential backup file and a differential backup management file in separate HiRDB file systems.
5. When the differential backup facility is used, the method for specifying the differential backup file name in the `-b` option depends on the specification of the `-k` option.

- If `-k u`, `-k i`, or `-k o` is specified, you must specify a different name for each differential backup file.
- This is not applicable when `-k e` or `-k m` is specified (the differential backup facility cannot be used).
- If `-k n` is specified, there is no need to specify a different policy name for each differential backup file (different names may be specified, in which case the operation becomes complex because a policy needs to be created for each differential backup file).

(22) -d backup-type

If you are using the differential backup facility, use this option to specify the type of backup to be made:

a: Makes a full backup.

b: Makes an accumulation-differential backup from the most recent full backup.

c: Makes an accumulation-differential backup from one of the following backups, whichever is most recent:

- Previous accumulation-differential backup
- Previous full backup

d: Makes a differential backup from the previous backup.

(23) -K HiRDB-file-system-area-name-storing-differential-backup-management-file

~ <pathname>

If you are using the differential backup facility, use this option to specify the name of the HiRDB file system area where the differential backup management file is to be stored. The utility uses the differential backup group name specified in the -g option as the name of the differential backup management file.

After executing pdcopy, you need to back up the differential backup management file using the pdfbkup command. If an error occurs in the differential backup management file, you must restore the differential backup management file from this backup copy using the pdfrstr command.

If you have executed pdmod while using the differential backup facility, make a full backup. In this case, also create a differential backup management file by specifying the -K option.

Make sure that "*name-of-HiRDB-file-system-area-storing-differential-backup-management-file*/*differential-backup-group-name*" does not exceed 167 characters.

(24) -o differential-backups-history-filename

~ <pathname>

If you are using the differential backup facility, use this option to specify the name of the file to which historical information about differential backups is to be output.

For details about historical information, see Section *18.6 Database copy utility process results listing*.

(25) -L differential-backup-management-file-size

~ <unsigned integer> ((1-2,046)) <<1>>

If you are using the differential backup facility, use this option to specify the size of the differential backup file in MB.

This size must be less than the value used when the HiRDB file system area was created (value of the `-n` option in the `pdfmkfs` command minus 5).

You can increase the size of this file a maximum of 23 times. The utility uses this value when increasing the size of the file.

(26) -q generation-number

~ <unsigned integer> ((0-10))

Specifies the generation number of the RDAREA to be backed up when the inner replica facility is used. When 0 is specified, the utility backs up the original RDAREA; when a value in the range 1-10 is specified, the utility backs up the specified replica RDAREA generation.

Rules

1. Specify this option together with the `-r` option. This option cannot be specified together with the `-a`, `-u`, or `-s` option.
2. Specify this option to back up the replica RDAREA with the indicated generation number in the inner replica group to which the original RDAREA specified in the `-r` option belongs.

(27) -S {backup-file-initial-size,increase-value | '(backup-file-initial-size, increase-value) [,(backup-file-initial-size,increase-value)]...'}

~ <unsigned integer> ((1-1048574)) <<100,10>>

This option is applicable to the backup file for which `i` is specified in the `-k` option.

Specify in MB an initial size and a size increase value for the backup file (the specified increase value is allocated when the amount of backup data exceeds the initial size).

Rules

1. The initial size must be less than the value specified in the `-n` option of the `pdfmkfs` command when the HiRDB file system area was created. This is because an area for system management information is required.
2. When this option is omitted, the utility assumes 100 MB as the backup file initial size and 10 MB as the increase size.
3. You can increase the size of a backup file a maximum of 23 times, provided that the number of increases does not exceed the permitted maximum number of increases specified when the HiRDB file system area was created.
4. If specifying multiple files, you can specify an initial allocation size and increase value for each file. In this case, enclose each set of initial allocation sizes and increase values in parentheses, then enclose the entire specification in single quotation marks. If you are specifying this information in the control statement file, do not use single quotation marks.
5. If the number of initial allocation sizes paired with increase values is less

than the number of specified files, the utility assumes the last pair of values for the remaining files. If the number of initial allocation sizes paired with increase values is more than the number of specified files, the utility ignores the excess pairs of values.

Example

```
-b file1,file2,file3,file4
-k i
-s '(10,1),(1,1)'
```

File	Initial size	Increase value	Remarks
`file1`	10	1	Set by the `(10, 1)` specification.
`file2`	1	1	Set by the `(1, 1)` specification.
`file3`	1	1	Because there is no specification, the last specified values `(1, 1)` are applied.
`file4`	1	1	

18.5 Notes

1. Do not back up an erroneous RDAREA. A backup copy of an erroneous RDAREA is invalid and cannot be used to restore the database.
2. To use EasyMT and MTguide, you need to initialize the magnetic tape before using this utility.
3. If both EasyMT and JP1/Magnetic Tape Access are installed, the utility starts JP1/Magnetic Tape Access.
4. The maximum number of copies of the database copy utility that can be executed concurrently depends on the value of the `pd_utl_exec_mode` operand in the system common definitions. If `pd_utl_exec_mode=0` is specified, you can execute a maximum of 32 copies; if `pd_utl_exec_mode=1` is specified, you can execute as many copies as there are specified in the `pd_max_users` operand. The number of copies of the database copy utility that can be executed concurrently depends on the number of specified backup files. The following describes the concept of concurrently executing database copy utility copies:

   Concept

   - If six copies of database copy utility are executed concurrently, each of which uses a single backup file, the number of concurrently executing database copy utility copies is 1 × 6 = 6.
   - If four copies of database copy utility are executed concurrently, each of which uses two single backup files, the number of concurrently executing database copy utility copies is 2 × 4 = 8.
   - If all the above database copy utilities are running concurrently, the number of concurrently executing utility copies is 6 + 8 = 14.

5. If you specify multiple backup files in the control statement file using a regular expression in the `-r` option, make sure that the RDAREAs subject to backup processing are output to a single backup file. An error results if they are output to multiple backup files. The following shows examples:

   Example 1

   `-b` *back-up-file-1* `-r RDUSER*,RDMASTER`

   `-b` *back-up-file-2* `-r ..USER,RDDICTIONARY`

   Explanation:

   An error results because the RDAREAs with names beginning with `RDUSER` are output to backup files 1 and 2.

   Example 2

`-b` *back-up-file-1* `-r RD*,RDMASTER`

Explanation:

This specification does not result in an error because there is only one backup file, although `RD*` includes `RDMASTER`.

6. The return codes of `pdcopy` are as follows:

   `0`: Normal termination

   `8`: Abnormal termination (some of the copy operations failed or were skipped)

   `12`: Abnormal termination (all copy operations failed)

7. You can copy a maximum of 1,120 RDAREAs in a single execution of `pdcopy`. If you have specified multiple backup files in the `-b` option in the control statements file, you can copy a maximum of 1,120 RDAREAs per backup file.

8. If you selected `utf-8` as the character encoding in the `pdsetup` command, you can use a file with a BOM as the control statements file for `pdcopy`. Note that even when a file with a BOM is used as the control statements file, the BOM is skipped. No BOM is included in the file that is output by `pdcopy`.

18.6 Database copy utility process results listing

(1) Contents of process results output file

The following shows the contents of the process results output files for the database copy utility.

```
pdcopy (VV-RR)  ***** DB COPY *****   1996-04-24 14:08:20 [1] utl3 [2]
-----------------------------------------------------------------
*** DB BACKUP INFORMATION LIST ***       BACKUPMODE : SHARED [3]
<<LOG FILE INFORMATION>>
  SYSTEM ID      : utl13 [4]
  UNIT ID        : sds1  [5]
  TYPE           : sys   [6]
  SERVER NAME    : sds   [7]
  SERVER RUN ID  : 455d67c7 [8]
  LOG SERVER RUN ID : 455d67c7 [9]
  FILE NAME      : logfg01 [10]
  GENERATION NO  : 1 [11]

  BLOCK NO       : d2 [12]
  HEADER WRITE COUNTER : 3 [13]
  HEADER WRITE TIME : 455d68d2 [14]
<<BACKUP FILE INFORMATION>>
  VOL NAME       : vol1,vol2 [15]
  FILE NAME      : host1:/usr/bkup01 [16]
                         /usr/bkup02
  FILE KIND      : u [17]
  STARTED AT     : 1996-04-24 14:08:20 [18]
  ENDED AT       : 1996-04-24 14:08:21 [19]

 <REPLICA BACKUP INFORMATION>
  REPLICA BACKUP GENERATION NO : 0 [41]

<<RDAREA INFORMATION>>
  UNIT NAME      :un16 [20] SERVER NAME : sds [21]

  RDAREA NAME    :USER1[22][23]           [24]                      [25]
  RDAREA ID      :          4 ATTRIBUTE : USER          PAGE SIZE : 4096
  STARTED AT     : 1996-04-24 14:08:20 [26]
  ENDED AT       : 1996-04-24 14:08:21 [27]
 <REPLICA RDAREA INFORMATION>
  ORIGINAL RDAREA NAME : USER1 [42]
  REPLICA RDAREA GENERATION NO : 0 [43]
 <RDAREA STATUS>      [28]
 TOTAL PAGE:           200 USED PAGE(RATE)   :        3   ( 1.5%) [29]
                           FREE SPACE        :       10.8         [30]
                           UNUSED PAGE(RATE) :      197   (98.5%) [31]
                               [32]
  INDEX PAGE(RATE) :            1( 33.3%) TOTAL SPACE :  4.0k      [33]
                           FREE SPACE(RATE)  :        3.8k( 94.7%) [34]
                               [35]
  DATA PAGE(RATE)  :            2( 66.7%) TOTAL SPACE :  8.0k      [36]
                           FREE SPACE(RATE)  :        7.0k( 87.6%) [37]


 <FILE INFORMATION>
 1 /dbarea/area1/rdmt06 [38]
   EXTENT COUNT :   1 [40]
 2 /dbarea/area1/rdmt07
   EXTENT COUNT :   1


<<RDAREA INFORMATION>>
  UNIT NAME   : un16        SERVER NAME : sds

  RDAREA NAME : USER2
  RDAREA ID   :            5 ATTRIBUTE  : USER             PAGE SIZE : 4096
  STARTED AT  : 1996-04-24 14:08:21
  ENDED AT    : 1996-04-24 14:08:21


 <REPLICA RDAREA INFORMATION>
  ORIGINAL RDAREA NAME : USER2
  REPLICA RDAREA GENERATION NO : 0
 <RDAREA STATUS>
 TOTAL PAGE:           200 USED PAGE(RATE)   :        3   ( 1.5%)
                           FREE SPACE        :       10.8
                           UNUSED PAGE(RATE) :      197   (98.5%)
  INDEX PAGE(RATE) :            1( 33.3%) TOTAL SPACE :  4.0k
                           FREE SPACE(RATE)  :        3.8k( 94.7%)
  DATE PAGE(RATE)  :            2( 66.7%) TOTAL SPACE :  8.0k
                           FREE SPACE(RATE)  :        7.0k( 87.6%)


 <FILE INFORMATION>
 1 /dbarea/area1/rdmt08
   EXTENT COUNT :   2
 2 /dbarea/area1/rdmt09
   EXTENT COUNT :   2
```

```
<<RDAREA INFORMATION>>
  UNIT NAME   : un16        SERVER NAME : sds

  RDAREA NAME : USER3
  RDAREA ID   :            6 ATTRIBUTE  : USER            PAGE SIZE : 4096
  STARTED AT  : 1996-04-24 14:08:21
  ENDED AT    : 1996-04-24 14:08:21

 <REPLICA RDAREA INFORMATION>
  ORIGINAL RDAREA NAME : USER3
  REPLICA RDAREA GENERATION NO : 0
 <RDAREA STATUS>
  TOTAL PAGE:        200 USED PAGE(RATE)   :          3   (  1.5%)
                         FREE SPACE        :         10.8
                         UNUSED PAGE(RATE) :        197   ( 98.5%)
  INDEX PAGE(RATE) :          1( 33.3%) TOTAL SPACE :    4.0k
                         FREE SPACE(RATE)  :          3.8k( 94.7%)
  DATA PAGE(RATE)  :          2( 66.7%) TOTAL SPACE :    8.0k
                         FREE SPACE(RATE)  :          7.0k( 87.6%)


 <FILE INFORMATION>
 1 /dbarea/area1/rdmt10
   EXTENT COUNT :   2
 2 /dbarea/area1/rdmt11
   EXTENT COUNT :   2


<<RDAREA INFORMATION>>
  UNIT NAME : un16     SERVER NAME : sds

  RDAREA NAME : USER_LOB1
  RDAREA ID   :         31  ATTRIBUTE : USER_LOB       PAGE SIZE : 8192
  STARTED AT  : 1996-04-24 14:08:21
  ENDED AT    : 1996-04-24 14:08:21


 <REPLICA RDAREA INFORMATION>
  ORIGINAL RDAREA NAME : USER_LOB1
  REPLICA RDAREA GENERATION NO : 0
 <RDAREA STATUS>
  TOTAL PAGE:     75000  USED PAGE(RATE)   :      30000   ( 40.0%)
                         UNUSED PAGE(RATE) :      45000   ( 60.0%)

  DISORDERED LOB DIRECTORY RATIO :   0% [39]


 <FILE INFORMATION>
 1 /dbarea/area1/rdmt12
   EXTENT COUNT :   2
 2 /dbarea/area1/rdmt13
   EXTENT COUNT :   2
```

Explanation

1. Date and time the database copy utility was executed (in the format *YYYY*/*MM*/*DD* *hh*:*mm*:*ss*).

   YYYY: Year. *MM*: Month. *DD*: Date. *hh*: Hour. *mm*: Minute. *ss*: Second.
2. HiRDB identifier
3. Backup acquisition mode:

   `EXCLUSIVE`: `-M x` or `-M r`

   `SHARED`: `-M s`
4. HiRDB identifier
5. Unit identifier
6. Log type:

   `sys`: System log file
7. Server name
8. Server run ID (8 hexadecimal digits)
9. Log server run ID (8 hexadecimal digits)
10. Name of the system log file needed to restore the database, together with the corresponding backup file.
11. Generation number of the system log file in 10.
12. Beginning block number (up to 8 hexadecimal digits)
13. Number of times header has been updated (up to 8 hexadecimal digits)
14. Usage start time (8 hexadecimal digits)
15. Volume name

    When e or m is specified in the `-k` option, the utility displays the value specified in the `-v` option; if the `-v` option was omitted, the utility displays `******`.
16. Backup file name (displayed as *host-name*:*path-name**).

    If multiple file names were specified, the utility displays only the names of those files that were used.

    If `-k e` or `-k m` was specified, the utility displays the value specified in the `-N` option. If the `-N` option was omitted, the utility displays `******`.

    If `-k o` was specified, the header becomes `OBJECT NAME`, not `FILE NAME`. In this case, the utility displays the object name of JP1/OmniBack II.

If `-k n` was specified, the header becomes `POLICY NAME`, not `FILE NAME`. In this case, the utility displays the policy name specified with NetBackup.

* If the system switchover facility is used, the utility displays the host name of the primary system.

17. Type of backup file (value specified in the `-k` option)
18. Backup start time (same format as in 1)
19. Backup end time (same format as in 1)
20. Name of the unit containing the RDAREA
21. Name of the server containing the RDAREA
22. Name of the copied RDAREA
23. Number of the copied RDAREA
24. Type of RDAREA:

    `MASTERDIRECTORY`: Master directory RDAREA

    `DATADIRECTORY`: Data directory RDAREA

    `DATADICTIONARY`: Data dictionary RDAREA

    `SYSTEM_LOB`: Data dictionary LOB RDAREA

    `USER`: User RDAREA

    `USER_LOB`: User LOB RDAREA

    `REG`: Registry RDAREA

    `REG_LOB`: Registry LOB RDAREA
25. Page length of the RDAREA (in bytes)
26. Backup start time for the RDAREA (same format as in 1)
27. Backup end time for the RDAREA (same format as in 1)
28. Number of available pages in the RDAREA (number of pages in the RDAREA where a table or index can be stored)
29. Number of used pages (number of pages in the RDAREA that already contains a table or index), and the percentage ratio of the number of used pages to the number of available pages
30. Size of free space in the pages used (in bytes)

    This information is displayed in the following format:

    ZZZ9 (applicable from 0 to less than 1 KB)

ZZZ9.9K (applicable from 1 KB to less than 1 MB)

ZZZ9.9M (applicable from 1 MB to less than 1 GB)

ZZZ9.9G (applicable to 1 GB or greater)

Note: Z is omitted if the value is `0`; 9 is never omitted, even if the value is `0`.

31. Number of unused pages (of all available pages in the RDAREA, the number of pages that has not yet been used), and the ratio of the number of unused pages to the number of available pages
32. Number of index storage pages (of all the used pages, the number of pages used to store an index), and the percentage ratio of the number of index storage pages to the number of used pages
33. Total size of the index storage pages (in bytes)
34. Total size of the free space in the index storage pages (in bytes), and the ratio of this size to the total size of the index storage pages
35. Number of table storage pages (of all the used pages, the number of pages used to store a table), and the percentage ratio of the number of table storage pages to the number of used pages
36. Total size of table storage pages (in bytes)
37. Total size of the free space in the table storage pages (in bytes), and the ratio of this size to the total size of the table storage pages
38. Names of the HiRDB files constituting the RDAREA
39. Access performance deteriorates if the system information disorganization ratio for a data dictionary LOB RDAREA or user LOB RDAREA is 1% or greater. In this case, reorganize the RDAREA using the database reorganization utility. For a data dictionary LOB RDAREA, reorganize the dictionary tables related to stored procedures (`SQL_ROUTINES`, `SQL_ROUTINE_RESOURCES`, and `SQL_ROUTINE_PARAMS`).
40. Number of HiRDB file extents
41. Generation number specified in the `-q` option
42. Name of original RDAREA
43. Generation number of replica RDAREA

Note

`<<LOG FILE INFORMATION>>` is displayed when `s` is specified in the `-M` option or when the `-z` option is specified.

Nos. 4-6, 8, 9, and 12-14 are displayed when the `-z` option is specified.

Nos. 42 and 43 are displayed when the inner replica facility is used.

(2) Contents of the history file for differential backups

The following shows the contents of the differential backups history file provided by the database copy utility:

```
 pdcopy (VV-RR) ***** DB COPY *****  2000-10-26 21:30:20[1] utl3[2]

 /hirdb/pdcopy/admfile[3]
 backupg1[4]
 2000-10-19 21:00:20[5]
 2000-10-26 21:18:20[6]
 RDAREA NAME :rdarea01,rdarea02[7]
[8]         [9]                 [10]    [11][12]          [13]
 a 2000-10-19 21:08:20,2000-10-19 21:09:25 r u host01: /hirdb/pdcopy/backup/backup01
 d 2000-10-20 21:01:20,2000-10-20 21:01:38 r u host01: /hirdb/pdcopy/backup/backup02
 c 2000-10-21 21:05:20,2000-10-21 21:06:10 r u host01: /hirdb/pdcopy/backup/backup03
 b 2000-10-22 21:01:20,2000-10-22 21:01:45 r u host01: /hirdb/pdcopy/backup/backup04
```

Explanation

1. Date and time the `pdcopy` command was executed
2. HiRDB identifier
3. Name of the HiRDB file system area containing the differential backup management file
4. Backup group name
5. Date and time the backup group was created
6. The last update date and time for the backup group
7. Names of the RDAREAs belonging to the backup group
8. Type of backup:

   `a`: Full backup

   `b`: Accumulation-differential backup from the most recent full backup

   `c`: Accumulation-differential backup from the previous accumulation-differential backup or previous full backup, whichever is more recent

   `d`: Differential backup
9. Backup start date and time
10. Backup end date and time
11. Backup acquisition mode:

`x`: Reference/update-impossible mode

`r`: Reference-possible mode

`s`: Updatable mode

12. Type of backup file

    `u`: Regular File

    `i`: HiRDB file system area for backup files

    `o`: JP1/OmniBack II object

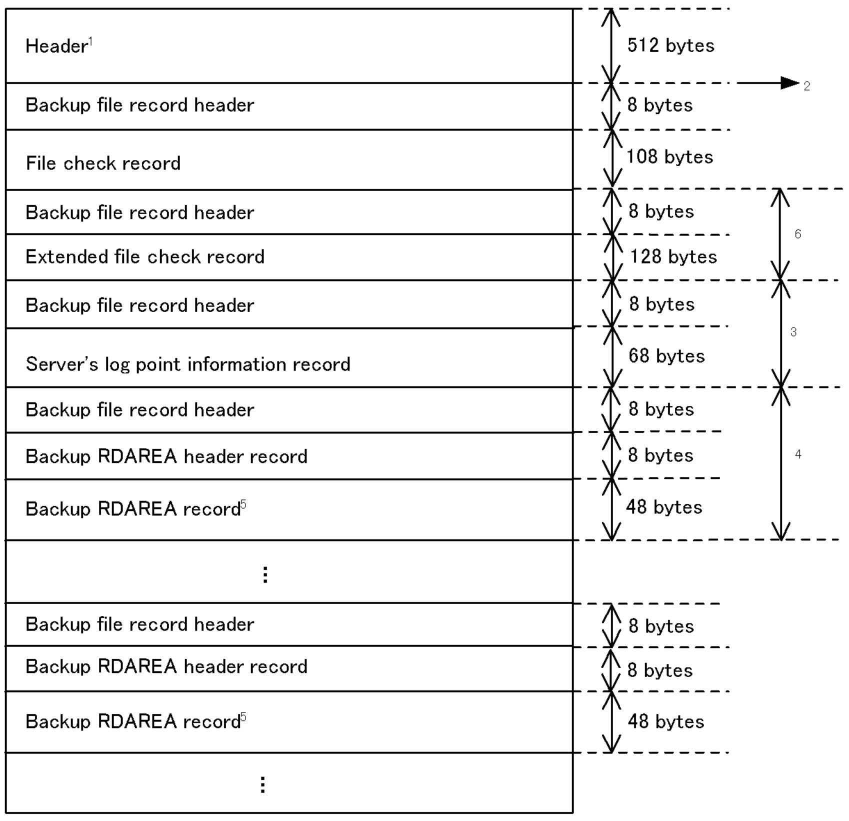
13. Backup file name or object name

    The backup file name is displayed in the format *host-name*:*backup-filename*. The object name is followed by (G barlist-filename).

18.7 Backup file format

Figures 18-3 to 18-7 show the formats used in the database backup file that is created by the database copy utility. Information in these backup files is used to manage backup generations, such as when a backup management tool is created. These backup file formats are applicable to a HiRDB/Single Server only.

Figure 18-3: Structure of the database backup file


Header[1]
512 bytes
2
Backup file record header
8 bytes
File check record
108 bytes
Backup file record header
8 bytes
6
Extended file check record
128 bytes
Backup file record header
8 bytes
3
Server's log point information record
68 bytes
Backup file record header
8 bytes
Backup RDAREA header record
8 bytes
4
Backup RDAREA record[5]
48 bytes
⋮
Backup file record header
8 bytes
Backup RDAREA header record
8 bytes
Backup RDAREA record[5]
48 bytes
⋮

Note: If multiple volumes are used, JP1/Magnetic Tape Library is required.

[1] When JP/Magnetic Tape Access is used, data is read in units of 32 KB using this program's interface (Easy MT label). In this case, headers are not provided. If JP1/Magnetic Tape Access is not used, data is read in units of 32 KB, in which case headers are provided.

[2] If data is backed up to a streaming tape (such as DAT) without using JP1/Magnetic Tape Access, file markers are inserted. If such a file marker is read, `0` is returned as its length.

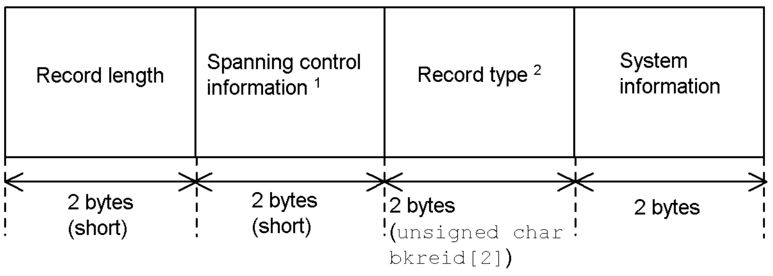
[3] This information is output only when the `-z` option is specified with the database copy utility. If the `-z` option is omitted, all subsequent records are shifted up.

[4] If the length exceeds 32,768 bytes, all records outside this range become spanned records. In this case, the 32,768-byte boundary is followed by the backup file record header, backup RDAREA header record, and backup RDAREA records.

[5] As many backup RDAREA records are placed as there are RDAREAs in the backup RDAREA header record.

[6] This is applicable only when the differential backup facility is used.

Figure 18-4: Data format of the backup file record header



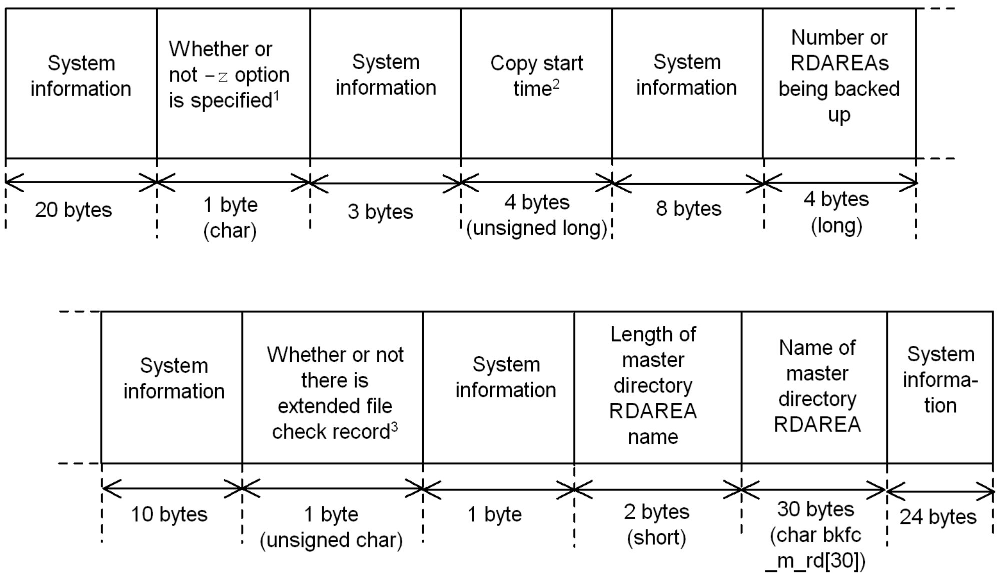
1 Spanning control information:

- `1`: Start of spanned record
- `2`: Intermediate spanned record
- `3`: End of spanned record
- `0`: Not a spanned record

2 Record type:

- `fc`: File check record
- `rd`: Backup RDAREA header record and backup RDAREA record
- `sv`: Server's log point information record
- `f2`: Extended file check record

Figure 18-5: Data format of the file check record



1 If the `-z` option is specified, `z` is set; otherwise, `0x00` is set.
2 Julian date from 1970-01-01 is set in hexadecimal (example: `3372bfb3`).
3 If there is an extended file check record, `0x01` is set.

Figure 18-6: Data format of the backup RDAREA header record

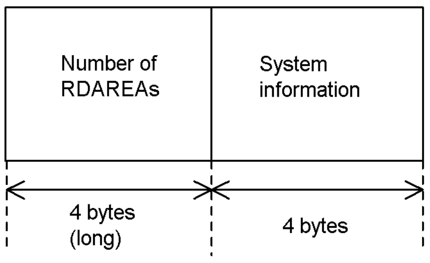
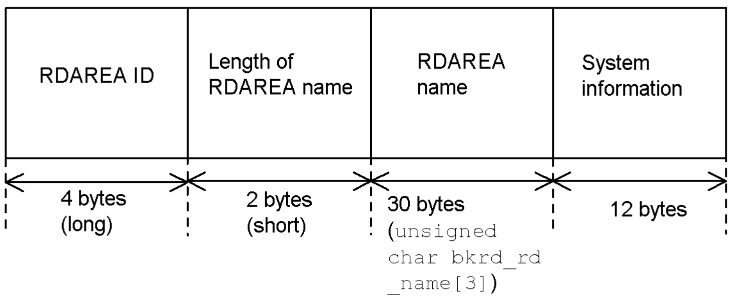


Figure 18-7: Data format of the backup RDAREA record



Chapter

19. Database Recovery Utility (pdrstr)

This chapter explains how to use the database recovery utility (`pdrstr`).

This chapter contains the following sections:

19.1 Functions of the database recovery utility

Executer: HiRDB

The database recovery utility provides the following functions:

- Database recovery
- Re-creation of log point information file

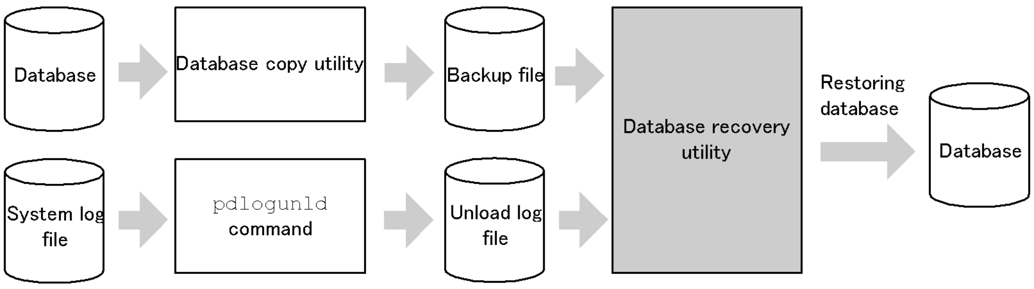
19.1.1 Database recovery

Whenever a database needs to be restored due to a disk error, etc., use the database recovery utility. In this case, you need the following information:

- Backup file
- Unload log file (or system log file if the HiRDB system is run without unloading the system log)

Figures 19-1 and 19-2 show an overview of database recovery.

Figure 19-1: Overview of database recovery (operation involving unloading of the system log)



Explanation

You can restore the database to the following status:

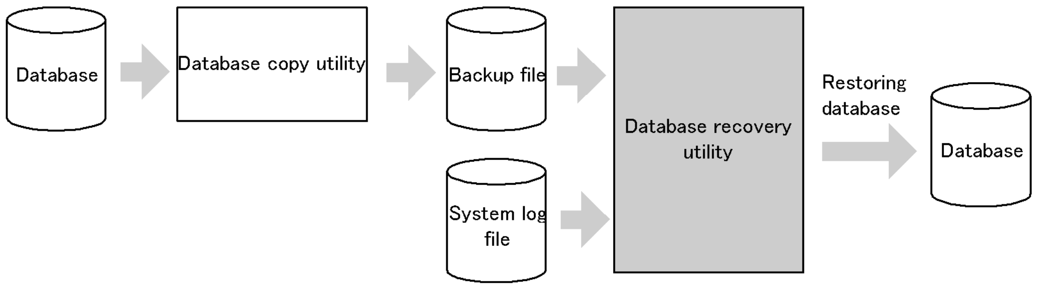
- **Status existing when the last backup was made**

  To restore the database to the status existing when the last backup was made, you only need the backup file. No unload log file is needed.

- **Most recent status (most recent synchronization point)**

  To restore the database to the most recent status, you need the backup file and the unload log file that was created by unloading the system log after the last backup processing.

Figure 19-2: Overview of database recovery (operation without unloading the system log)



Explanation

You can restore the database to the following status:

- **Status existing when the last backup was made**

  To restore the database to the status existing when the last backup was made, you only need the backup file. No system log is needed.

- **Most recent status (most recent synchronization point)**

  To restore the database to the most recent status, you need the backup file and the system log file that contains the system log output after the last backup processing.

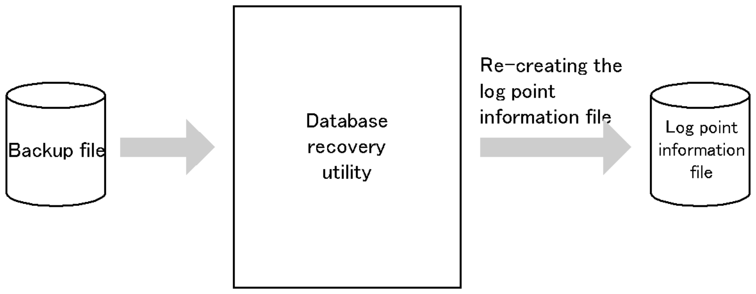
19.1.2 Re-creating a log point information file

If running the HiRDB system without unloading system log, you need to store a log point information file. If the log point information file is damaged due to disk errors, etc., re-create it using the database recovery utility. To do this, you need the following input information:

- Backup file

Figure 19-3 shows an overview of re-creation of a log point information file.

Figure 19-3: Overview of re-creation of a log point information file


Backup file
Database recovery utility
Re-creating the log point information file
Log point information file

19.2 Information required for recovering databases

(1) Database recovery units

You can restore a database in the following units. The database recovery utility provides an option to specify these recovery units. Table 19-1 shows the database recovery units.

Table 19-1: Database recovery units

Database recovery unit	Description	Option
By the system	Restores all RDAREAs in the system (including master directory RDAREAs), except the user RDAREAs.	`-a`
By the backup file	Restores the RDAREAs in the backup file.	`-c`
By the unit*	Restores all RDAREAs in a specified unit.	`-u`
By the server*	Restores all RDAREAs in a specified server.	`-s`
By the RDAREA	Restores specified RDAREAs. A group of RDAREAs can be restored by specifying their RDAREA names as a regular expression.	`-r`

* This is applicable to a HiRDB/Parallel Server only, not applicable to a HiRDB/Single Server.

(2) Status of RDAREAs subject to recovery processing

The RDAREAs subject to recovery processing (except the master directory RDAREA) must be in shutdown and closed status (`CLOSE HOLD(CMD)` or `CLOSE HOLD` displayed by the `pddbls` command).

(3) Relationship with the backup acquisition mode

If the backup file was created by the database copy utility specifying the `-M s` option, you need not only the backup file but also the unload log file or system log file.

(4) Restoring the master directory RDAREAs

To restore an RDAREA, you need to place the RDAREA in the shutdown and closed status by the `pdhold` command. However, the master directory RDAREA cannot be placed in the shutdown and closed status. To restore the master directory RDAREA, you need to terminate HiRDB once and then restart it using the `pdstart -r` command.

Therefore, to restore all RDAREAs including the master directory RDAREA, you must start HiRDB using the `pdstart -r` command.

(5) Restoring only backup data using the backup file acquired before a configuration change

To restore only backup data using the backup file acquired before the configuration was changed (HiRDB file was added), first delete all the HiRDB files constituting the RDAREA.

(6) Restoring to the most recent status

To restore an RDAREA to the most recent status, you need the backup file and the system log file that is output after the last backup. This means that you need the system log stored in the current system log file. Because the current system log file cannot be unloaded, use the `pdlogswap` command to swap the log files, then unload the file containing the current system log information.

(7) Database restored using the unload log file or system log file

If you have restored a database using an unload log file or a system log file, be sure to back up the restored RDAREA immediately after recovery processing. If you do not and an error occurs, the database recovery utility cannot restore the database to the synchronization point immediately before the error.

To restore a database using unload log files, you must specify all required unload log files. If any required unload log files are missing, the utility displays a message (`KFPR16203-E` or `KFPR16301-E`) and terminates with an error.

(8) Recovery when operating HiRDB without unloading the system log

If you are operating HiRDB without unloading system log information, then in the event of a database error you can recover the database using the system log information as the input information for the database recovery utility, without having to unload the system log information. This mode of operation is called *operation without unloading the system log*.

Operation without unloading the system log is based on the log point concept. To recover a database from an error, you do not need the system log information from before the point when the last backup was made. The location that distinguishes the system log information required for recovery of the database from the unneeded system log information is called a *log point*. A log point is set when the database copy utility is used to make a backup copy.

To recover the database during operation without unloading the system log, specify the `-L` and `-z` options.

For details about operation without unloading the system log, see the *HiRDB Version 8 System Operation Guide*.

(9) Recovery from a backup using the differential backup facility

To recover the database from its backup obtained using the differential backup facility, specify the `-g` and `-K` options.

For details about the differential backup facility, see the *HiRDB Version 8 System Operation Guide*.

(10) Recovery from a backup using the inner replica facility

To recover the database from its backup obtained using the inner replica facility, specify the `-q` and `-x` options.

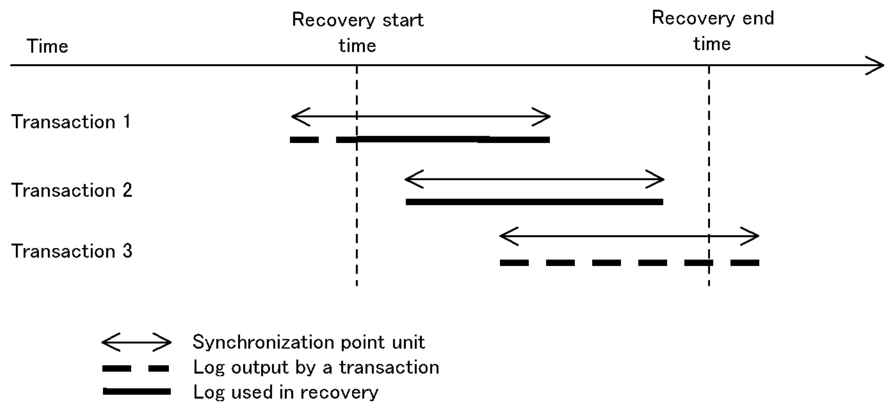
For details about the inner replica facility, see the *HiRDB Staticizer Option Version 7 Description and User's Guide*.

(11) Recovery with range specification

When restoring a database using an unload log file or system log file, you can specify a range of recovery time. This time range can also be specified for each RDAREA to be recovered. When a time range is specified, only those logs output by transactions with synchronization points that fall within the specified time range are recovered. Following is an example of logs subject to recovery when the database is restored with a time range specified. For details about recovery with a time range specification, see the *HiRDB Version 8 System Operation Guide*.

Figure 19-4 shows an overview of recovery with a time range specified.

Figure 19-4: Overview of recovery with a time range specified



Explanation

If a transaction is executing at the start point, such as Transaction 1 in the figure, the utility recovers it using the log collected since the start point.

The utility does not recover a transaction, such as Transaction 3, that has not reached a synchronization point by the recovery end time. The utility recovers a transaction that has reached a synchronization point within the specified time range, using the log up to the synchronization point. For a transaction, such as Transaction 3, that has not reached a synchronization point, the utility issues a warning message to that effect.

(12) HiRDB status during database recovery utility execution

1. You can execute the database recovery utility only when HiRDB is active.
2. Execute the database recovery utility at the server machine containing the single server or the server machine where the system manager is located.
3. To execute the database recovery utility, the unit containing the RDAREAs subject to recovery processing must be active, as well as the unit containing the unload log file, system log file, or backup file used for recovery processing. If only a backup file is used to restore RDAREAs, the server containing the RDAREAs to be restored may not be active; however, if an unload log file or system log file is to be used, the server must be active.

19.3 Examples

This section presents examples of database recovery and log point information file re-creation. The topics include:

- Restoring in units of systems
- Restoring in units of RDAREAs
- Restoring in units of servers
- Restoring RDAREAs using JP1/OmniBack II
- Restoring from a backup using the differential backup facility
- Restoring RDAREAs from their backups using NetBackup
- Re-creating the log point information file

For details about the `pdrstr` command options used in these examples, see Section *19.4 Command format*.

19.3.1 Restoring in units of systems

(1) Restoring all RDAREAs

This example restores all RDAREAs to the point when the last backup was made:

```
pdrstr -m /hirdb/rdarea/mast/mast01 -b /usr/hirdb/pdcopy/backup/backup01
-p /usr/hirdb/pdrstr/list/list01 -a
```

Explanation

`-m`: Specifies the name of the first HiRDB file in the master directory RDAREA.

`-b`: Specifies the name of the backup file.

`-p`: Specifies the output file name for the database recovery utility process results listing.

`-a`: Specifies that all RDAREAs are to be restored.

(2) Restoring all RDAREAs from a backup on DAT

This example restores all RDAREAs to the point when the last backup was made, using the backup made to a file on DAT.

```
pdrstr -m /hirdb/rdarea/mast/mast01 -b /dev/rmt/0m
-p /usr/hirdb/pdrstr/list/list01 -a
```

Explanation

-m: Specifies the name of the first HiRDB file in the master directory RDAREA.

-b: Specifies the name of the backup file.

-p: Specifies the output file name for the database recovery utility process results listing.

-a: Specifies that all RDAREAs are to be restored.

19.3.2 Restoring in units of RDAREAs

(1) Restoring in units of RDAREAs

This example restores two user RDAREAs (RDAREA01 and RDAREA02) to the most recent status:

```
pdrstr -m /hirdb/rdarea/mast/mast01 -b /usr/hirdb/pdcopy/backup/backup01
-l /usr/hirdb/pdlogunld/unld01 -p /usr/hirdb/pdrstr/list/list01 -w /tmp/sortwork
-r RDAREA01,RDAREA02
```

Explanation

-m: Specifies the name of the first HiRDB file in the master directory RDAREA.

-b: Specifies the name of the backup file.

-l: Specifies the name of the unload log file. This is the name of the unload log file that was created after the backup was made.

-p: Specifies the output file name for the database recovery utility process results listing.

-w: Specifies the work directory for sorting.

-r: Specifies that RDAREAs to be restored.

(2) Restoring RDAREAs during operation without unloading the system log

This example restores two user RDAREAs (RDAREA01 and RDAREA02) to the most recent status. It assumes that the HiRDB system is running without unloading the system log:

```
pdrstr -m /hirdb/rdarea/mast/mast01 -b /usr/hirdb/pdcopy/backup/backup01
-L -p /usr/hirdb/pdrstr/list/list01 -w /tmp/sortwork -r RDAREA01,RDAREA02
```

Explanation

-m: Specifies the name of the first HiRDB file in the master directory RDAREA.

-b: Specifies the name of the backup file.

-L: Specifies that the system log file is to be read.

-p: Specifies the output file name for the database recovery utility process results listing.

-w: Specifies the work directory for sorting.

-r: Specifies that all RDAREAs are to be restored.

19.3.3 Restoring in units of servers

This example restores all RDAREAs in a back-end server (bes1) to the most recent status. It assumes that a HiRDB/Parallel Server is running without unloading the system log:

```
pdrstr -m /hirdb/rdarea/mast/mast01 -b /usr/hirdb/pdcopy/backup/backup01
-L -p /usr/hirdb/pdrstr/list/list01 -w /tmp/sortwork -s bes1
```

Explanation

-m: Specifies the name of the first HiRDB file in the master directory RDAREA.

-b: Specifies the name of the backup file.

-L: Specifies that the system log file is to be read.

-p: Specifies the output file name for the database recovery utility process results listing.

-w: Specifies the work directory for sorting.

-s: Specifies that all RDAREAs in the back-end server (bes1) are to be restored.

19.3.4 Restoring RDAREAs using JP1/OmniBack II

(1) Restoring to the point when the last backup was made

This example restores all RDAREAs to the point when the last backup was made:

```
pdrstr -m /hirdb/rdarea/mast/mast01 -b host01:backup001 -p /usr/hirdb/pdrstr/list/
list01 -k o -G DLT01 -a
```

Explanation

-m: Specifies the name of the first HiRDB file in the master directory RDAREA.

-b: Specifies the name of the backup file (JP1/OmniBack II's object name).

-p: Specifies the output file name for the database recovery utility process results

-k: Specifies the type of backup file. This example specifies o as the type of backup file because it uses JP1/OmniBack II's object file.

-G: Specifies the name of the barlist file.

-a: Specifies that all RDAREAs are to be restored.

(2) Restoring to the most recent status

This example restores two user RDAREAs (RDAREA01 and RDAREA02) to the most recent status:

```
pdrstr -m /hirdb/rdarea/mast/mast01 -b host01:backup001
-l /usr/hirdb/pdlogunld/unld01 -w /tmp/sortwork -p /usr/hirdb/pdrstr/list/list01
-k o -G DLT01 -r RDAREA01,RDAREA02
```

Explanation

-m: Specifies the name of the first HiRDB file in the master directory RDAREA.

-b: Specifies the name of the backup file (JP1/OmniBack II's object name).

-l: Specifies the name of the unload log file. This is the name of the unload log file that was created after the last backup was made.

-w: Specifies the work directory for sorting.

-p: Specifies the output file name for the database recovery utility process results listing.

-k: Specifies the type of backup file. This example specifies o as the type of backup file because it uses JP1/OmniBack II's object file.

-G: Specifies the name of the barlist file.

-r: Specifies the RDAREAs to be restored.

19.3.5 Restoring from a backup using the differential backup facility

This example restores user RDAREAs (`rdarea01` and `rdarea02`) to the point where their differential backup was obtained.

```
pdrstr -m /rdarea/mast/mast01 -g backupg1 -K /pdcopy/admfile -r rdarea01,rdarea02
```

Explanation

-m: Specifies the name of the first HiRDB file in the master directory RDAREA.

-g: Specifies the name of the differential backup group.

-K: Specifies the name of the HiRDB file system area that stores the differential backup management file.

-r: Specifies the names of the RDAREAs to be restored (rdarea01 and rdarea02).

19.3.6 Restoring a specified RDAREA generation when the inner replica facility is used

This example restores a specified RDAREA generation when the inner replica facility is used.

```
pdrstr -m /hirdb/rdarea/mast/mast01 -b /bkdir/bkup02
-r MAST,DIR,DIC,USR01,USR02,USR03 -q 1
```

Explanation

-m: Specifies the name of the first HiRDB file in the master directory RDAREA.

-b: Specifies the name of the backup file.

-r: Specifies the RDAREAs to be restored (MAST, DIR, DIC, USR01, USR02, and USR03).

-q: Specifies the generation number of the RDAREAs to be restored.

19.3.7 Restoring RDAREAs from their backup using NetBackup

This example restores user RDAREAs (RDAREA01 and RDAREA02) to the point where their backup was made.

```
pdrstr -m /hirdb/rdarea/mast/mast01 -b host01:POLICY01 -k n -r RDAREA01,RDAREA02
```

Explanation

-m: Specifies the name of the first HiRDB file in the master directory RDAREA.

-b: Specifies the name of the backup file (NetBackup policy name).

-k: Specifies the type of backup file. This example specifies n to use NetBackup policy.

-r: Specifies the RDAREAs to be restored.

19.3.8 Re-creating the log point information file

This example re-creates the log point information file:

```
pdrstr -b /usr/hirdb/pdcopy/backup/backup01 -z /usr/hirdb/pdcopy/logpoint/logpoint01
```

Explanation

`-b`: Specifies the name of the backup file to be read.

`-z`: Specifies the name of the log point information file to be re-created.

19.4 Command format

19.4.1 Option format

The required or frequently specified options are indicated in bold. In the table below, each number corresponds to the number assigned to each option, which is explained in *19.4.2 Options*.

When separating an option's flag arguments with the comma, do not specify spaces before or after the comma. If a space is specified, the command ignores whatever is specified following the space.

(1) Database recovery

No.	Option
1	`pdrstr -m [`*host-name*`:]`*name-of-first-HiRDB-file-in-master-directory-RDAREA*

2	`[-b {[`*host-name*`:]`*backup-file-name*
	`[,`*backup-file-name*`]...`
	`|[`*host-name*`:]`*device-symbolic-name*`[,`*device-symbolic-name*`]`
	`|[`*host-name*`:]`*device-group-name*
	`|[`*host-name*`:]`*object-name*
	`|[`*host-name*`:]`*policy-name*`}]`

6	`[-p` *process-results-output-file-name*`]`
7	`[-w` *name-of-work-directory-for-sorting*`]`
8	`[-y` *work-buffer-size-for-sorting*`]`
9	`[-k {u|i|e|m|o|n}`
10	`[-E` *EasyMT-MT-attributes-definition-file-name*`]`
11	`[-B` *EasyMT-I/O-buffer-sectors-count*`]`
12	`[-v` *volume-name*`[,`*volume-name*`]...]`
13	`[-N` *EasyMT-file-name*`]`
14	`[-G` *barlist-file-name*`]`
15	`[-U {`*backup-search-condition-start-time*`,`*backup-search-condition-end-time* `|,`*backup-search-condition-end-time*`}]`
16	`[-g` *differential-backup-group-name*`]`
17	`[-K` *name-of-HiRDB-file-system-area-storing-differential-backup-management-file*`]`
18	`[-q` *generation-number*

No.	Option
19	`[-x` *generation-number*`]]`
20	`[-Y` *write-buffer-size*`]`
21	`[-f` *control-statement-file-name**`]`

* The options that can be specified in the control information file are shown as follows. You can also specify these options directly in the `pdrstr` command.

Contents of control statement file

No.	Option
3	`{{-l [`*host-name*`:]`*unload-log-file-name* `[,`*unload-log-file-name*`]...`
4	`|-L`
5	`|-d [`*host-name*`:]`*unload-log-file-storage-directory-name* `[,`*unload-log-file-storage-directory-name*`]...}`
22	`|{-a`
23	`|-c`
24	`|-u` *unit-identifier*`[,`*unit-identifier*`]...`
25	`|-s` *server-name*`[,`*server-name*`]...`
26	`|-r` *RDAREA-name*`[,`*RDAREA-name*`]...}`
27	`[-T{`*recovery-start-time*`,`*recovery-end-time*`|`*recovery-start-time*`|,`*recovery-end-time*`}]}`

(2) Re-creation of log point information file

No.	Option
2	**`pdrstr -b {[`*host-name*`:]`*backup-file-name* `|[`*host-name*`:]`*device-symbolic-name* `|[`*host-name*`:]`*device-group-name* `|[`*host-name*`:]`*object-name* `|[`*host-name*`:]`*policy-name*`}`**
9	**`[-k {`<u>`u`</u>`|i|e|m|o|n}]`**
10	`[-E` *EasyMT-MT-attributes-definition-file-name*`]`
11	`[-B` *EasyMT-input-output-buffer-sectors-count*`]`

No.	Option
12	[-v *volume-name*]
13	[-N *EasyMT-file-name*]
14	[-G *barlist-file-name*]
15	[-U {*backup-search-condition-start-time*,*backup-search-condition-end-time* |,*backup-search-condition-end-time*}]
28	**-z *log-point-information-file-name***

(3) Specifiability of options when another backup program is linked

Option		Other backup program		
		EasyMT	JP1/OmniBack II	NetBackup
-m		M	M	M
-b		M	M	M
-l		O	O	O
-L		O	O	O
-d		O	O	O
-p		O	O	O
-w		O	O	O
-y		O	O	O
-k	u	N	N	N
	i	N	N	N
	e	M[1]	N	N
	m	M[1]	N	N
	o	N	M	N
	n	N	N	M
-E		O	N	N
-B		O	N	N
-v		O	N	N

Option	Other backup program		
	EasyMT	JP1/OmniBack II	NetBackup
`-N`	O	N	N
`-G`	N	M	N
`-U`	N	N	O
`-g`	O	O	O
`-K`	O	O	O
`-q`	O	O	O
`-x`	O	O	O
`-y`	O	O	O
`-f`	O	O	O
`{-a|-c|-u|-s|-r}`	M[2]	M[2]	M[2]
`-T`	O	O	O
`-z`	O	O	O

Legend:

M: Mandatory (must be specified)

O: Optional (can be specified)

N: Not permitted (cannot be specified)

[1] You must specify either `e` or `m`.

[2] You must specify one of `-a`, `-c`, `-u`, `-s`, or `-r`.

19.4.2 Options

(1) -m [host-name:]name-of-first-HiRDB-file-in-master-directory-RDAREA

~ <identifier:pathname>

Specifies the name of the first HiRDB file in the master directory RDAREA. Specify the name of the host containing the master directory RDAREA and the path name, separated by a colon (:). This is the name specified in the `pd_master_file_name` operand in the system common definitions.

Example

```
-m host01:/hirdb_s/rdarea/rdmast
```

Rules

- You can omit the host name if the master directory RDAREA is located at the server machine where the database recovery utility is executed (where the `pdrstr` command is entered). In this case, specify only the path name.
- If you specify a host name, make sure that it is a host name specified in the `-x` option of the `pdunit` operand in the system definition.
- If you are using the system switchover facility, be sure to specify the host name of the primary system.

(2) -b {[host-name:]backup-file-name[,backup-file-name]...|[host-name:]device-symbolic-name[,device-symbolic-name]|[host-name:]device-group-name|[host-name:]object-name|[host-name:]policy-name}

～<identifier: path name> ((up to 167 characters when `-k i` is specified))

Specifies a backup file name, object name of JP1/OmniBack II, or a policy name of NetBackup.

When specifying a backup file name, device symbolic name, or device group name

Specify the name of the host that will contain the backup files, separated by a colon (:) from the names of the backup files, or from the device symbolic name, or from the device group name. Specify a pathname for a backup filename, and an identifier for a device symbolic name or device group name.

Example

```
-b host01:/usr/pdcopy/backup01
```

JP1/OmniBack II's object name specified

When using JP1/OmniBack II, specify the name of the host where JP1/OmniBack II's disk agent is located, separated by a colon from the object name.

Example

```
-b host01:backup001
```

When specifying a policy name of NetBackup

When using NetBackup, specify the name of the host that contains the NetBackup client, separated by a colon (:) from a policy name.

Example

```
-b host01:POLICY01
```

Rules

- If the backup file is located at the server machine where the database recovery utility is executed (where the `pdrstr` command is entered), you can omit the host name; in which case, specify the path name only.
- If you specify a host name, make sure that it is a host name specified in the `-x` option of the `pdunit` operand in the system definition.
- If the server machine used to execute the database recovery utility has a JP1/OmniBack II disk agent or a NetBackup client (where the `pdrstr` command is entered), you can omit the host name.
- If you are using the system switchover facility, make sure that the host name of the primary system is specified. Note, however, that the actual backup file, the JP1/OmniBack II disk agent, or the NetBackup client must on the running system.
- When re-creating the log point information file, specify only the first file or first volume, even if multiple backup files or volumes are to be used (the utility references only the first file or volume, even if multiple files or volumes are specified).
- If you specify multiple backup files that were acquired individually by RDAREA, only the first backup file specified is recognized. No other backup file is recognized.
- The following table shows the relationship between the `-b` and `-k` options:

-k option specification	Specifiable backup file
`-k u`	You can specify a regular file name or MT special file name. If a single backup was created in multiple files, specify the names of all such backup files in the order they were created.
`-k i`	You can specify a backup file in the format "*HiRDB-file-system-area-name*/*HiRDB-file-name*". If a single backup was created in multiple files, specify the names of all such backup files in the order they were created. Note that a HiRDB file system area name is not case-sensitive, but an HiRDB file name is case-sensitive.
`-k e`	You can specify an MT special file name.
`-k m`	You can specify a maximum of two device symbolic names or one device group name managed by MTguide. The system does not check whether a device symbolic name or a device group name is specified.
`-k o`	Specify the object name of JP1/OmniBack II that was obtained by `pdcopy`. For the host name, specify the name of the host that has both a disk agent of JP1/OmniBack II and HiRDB. HiRDB retrieves only the most recent object of JP1/OmniBack II.
`-k n`	Specify the policy name of NetBackup that was obtained by `pdcopy`. For the host name, specify the name of the host that has both NetBackup client and HiRDB. HiRDB retrieves only the most recent backup of NetBackup.

(3) -l [host-name:]unload-log-filename [,unload-log-filename]...

~ <identifier: pathname>

Specifies the names of unload log files created after the last backup file was created. Specify the name of the host containing the unload log files separated from the path names by a colon.

Example

```
-l host01:/usr/pdlogunld/unld01
```

Rules

- When the `-l` option is specified, the `-L`, `-c`, and `-d` options cannot be specified.
- If the backup file is located at the server machine where the database recovery utility is executed (where the `pdrstr` command is entered), you can omit the host name; in which case, specify the path name only.
- If you are using the system switchover facility, make sure that the host name of the primary system is specified. Note that the actual backup file must be located in the running system.
- If a regular file was used to create the unload log file, specify the name of that regular file as the unload log file name. If a HiRDB file was used to create the unload log file, specify the unload log file name in the format "*HiRDB-file-system-area-name*/*HiRDB-file-name*". Make sure that "*HiRDB-file-system-area-name*/*HiRDB-file-name*" does not exceed 167 characters.
- If the `pdstart -r` command was used to start the HiRDB, the `pdrstr` command with the `-l` option specified can recover only the master directory RDAREA.

Notes

- If you have specified the backup file for the `-b` option, created by the database copy utility specifying the `-M s` option, be sure to also specify the `-l` option. If there are multiple unload log files, you need to specify all of them in the chronological order of their generations, starting with the oldest one. In this case, specify the host name only once at the beginning. This means that all unload log files must be located at the specified host.
- To use unload log files for recovery, you must specify all required unload log files. If any required unload log file is missing, the utility issues a message (`KFPR16203-E` or `KFPR16301-E`) and terminates with an error.

(4) -L

When the system operation does not involve unloading of the system log, this option specifies that the system log files are to be input directly for database recovery.

Rules

- When you are specifying this option, you also need to specify the name of the backup file with the `-b` option. This backup file must have been created by the `pdcopy` command specifying the `-z` option.
- When this option is specified, the `-l`, `-c`, and `-d` options cannot be specified.

(5) -d [host-name:]unload-log-file-storage-directory-name[,unload-log-file-storage-directory-name]...

~ <identifier: path name>

Specifies the names of the directories that contain all the unload log files required for recovery. Separate the name of the host containing the unload log files and the path names with a colon (`:`).

Criteria

Use this option when many unload log files are needed for recovery (such as when there are too many unload log files to be specified on the command line for a single execution of `pdrstr`).

Rules

- Make sure that the total length of the name of an unload log file storage directory and the file names in the directory does not exceed 1,023 bytes. You can specify a maximum of 128 directories.
- Make sure that the specified directories do not contain any files other than unload log files. The utility ignores any file that is not an unload log file.
- This option cannot be specified together with the `-l` or `-L` option.
- When this option is specified, the utility reads the unload log files in the order they were created (in chronological order of the system log allocation times); therefore, make sure that an unload log file created before or after the machine time was changed is not mixed in in a directory.
- A HiRDB file system area cannot be specified in this option.
- The input start position of the unload log files in a specified directory depends on the backup acquisition mode of the backup file and the options specified during recovery. The following table describes the input start position of the unload log files:

Condition	Input start position of unload log files
A backup file acquired in the updatable mode is used.	The utility inputs all unload log files in the directory.
A backup file in the referencing-possible mode or in the reference/update-impossible mode is used.	Of the unload log files in the directory, the utility inputs all log information starting with the unload log file that includes the backup file acquisition start time. If there is any unload log file that needs to be input before the backup acquisition start time due to a time difference between machines, use the `-T` option to specify the recovery start time. Obtain the recovery start time based on the time difference between the backup destination host and the recovery target host.
The `-q` option was specified during backup acquisition and recovery, and the generation numbers specified in the `-q` option during backup acquisition and recovery do not match.	The utility inputs all unload log files in the directory.
`-x c` is specified during recovery.	
No backup file is used during recovery.	
The recovery start time is specified in the `-T` option during recovery.	The utility inputs the unload log files that fall on or after the recovery start time.

(6) -p process-results-output-file-name

~ <pathname>

Specifies the name of the file to which the processing result of the database recovery utility is to be output.

Example

```
-p /usr/pdrstr/list01
```

Rules

- Specify the path name of the server machine at which the database recovery utility is executed (where the `pdrstr` command is entered).
- If the specified path name is not found or this option is omitted, the database recovery utility outputs its processing result to the server machine where the utility is executed. The utility displays the output destination in the `KFPR26222-I` message.
- Regardless of the specification of this option, the utility outputs all error

messages to the system log file and standard output, and the final processing result to the standard output. However, error messages may not be output to the system log file and to the standard output in the same order.

(7) -w name-of-work-directory-for-sorting

~ <pathname>

When the `-l`, `-L`, or `-d` option is specified, use this option to specify the name of the directory under which the temporary file is to be created. If the RDAREAs subject to recovery processing are located on multiple server machines, the specified directory must be in each of the server machines. When this option is omitted, the utility assumes `/tmp`.

At least the following amount of free space must be available in the work directory for sorting (bytes):

$$(a + a/130 \times 36) \times 2$$

a: Value depends on the maximum amount of log information that is output per transaction:

- When the amount of log information does not exceed 1536000000 bytes

  min (total capacity of log file used per recovery, 1536000000)
- When the amount of log information exceeds 1536000000 bytes

  Total amount of log information output per transaction + 1536000000

(8) -y work-buffer-size-for sorting

~ <unsigned integer> ((256-2097152)) <<1024>>

When the `-l`, `-L`, or `-d` option is specified, specifies the size of the work buffer (in kilobytes). If this value is too small, the `KBLS300-E` message is displayed and the database recovery process terminates abnormally. Use the following guideline for determining an appropriate value:

$$b \text{ (kilobytes)} = \sqrt{(a+a \div 130 \times 36) \times 2}$$

Size of work buffer for sorting > *b* (kilobytes)

a: The value depends on the maximum amount of log information that is output per transaction.
- When the maximum amount does not exceed 1536000000 bytes
  min (total size of log files used for recovery), 1536000000)
- When the maximum amount exceeds 1536000000 bytes
  Maximum amount of log information output per transaction + 1536000000

(9) -k {u|i|e|m|o|n}

Specifies the type of backup file.

`u`

Specifies that a regular file backup or an MT backup is to be input without using EasyMT, JP1/OmniBack II, or NetBackup. If the backup file consists of multiple volumes, you cannot specify this option.

`i`

Specifies that the input is a backup file created in a HiRDB file system area. This option is applicable when the backup file is shared by the running system and a standby system, and the standby system is to be recovered using the backup made by the running system.

`e`

Specifies that the input is a magnetic tape backup file (EasyMT). You cannot specify this option if the backup file consists of multiple volumes. This specification requires the use of EasyMT.

`m`

Specifies that the input is a magnetic tape backup file using EasyMT and MTguide. You must specify this option if the backup file consists of multiple volumes. This specification requires the use of EasyMT and MTguide.

`o`

Specifies that the database is to be restored using JP1/OmniBack II. If you are specifying this option, specify JP1/OmniBack II's object name in the `-b` option.

`n`

Specifies that the database is to be recovered using NetBackup. In this case, specify the policy name of NetBackup in the `-b` option.

(10) -E EasyMT-MT-attributes-definition-file-name

~ <path name>

Specifies the name of the EasyMT MT attributes definition file. This file must be connected to the server machine where the database recovery utility is executed (where the `pdrstr` command is entered). The `-E` option is applicable to a backup file that has been created with e or m specified in the `-k` option. The following attributes take effect:

- `buffno`: Number of I/O buffer sectors
- `magazin`: MT device allocation pattern
- `job`: Job name

- `expire`: Expiration date
- `preserve`: Number of days to be saved

When the `-B` option is specified, it takes precedence over the number of I/O buffer sectors specified in the MT attributes definition file.

The utility checks the information specified in this file when EasyMT is executed.

(11) -B EasyMT-I/O-buffer-sectors-count

~ <unsigned integer> ((1-255))

Specifies the number of I/O buffer sectors to be used for magnetic tape input/output operations. A larger value results in better performance, but it requires more memory.

Rules

- The `-B` option is applicable to a backup file that has been created with e or m specified in the `-k` option.
- If the `-E` and `-B` options are both omitted, the utility assumes EasyMT's default values.

(12) -v volume-name[,volume-name]...

~ <alphanumerics> ((1-6 characters))

Specifies the volume names of the magnetic tapes that contain the backup data. When this option is specified, an error results if a volume mounted at a tape deck does not match any of the specified values. If the number of existing backup volumes is larger than the number of volumes specified here, the utility does not check any excess volume names. When this specification is omitted, the utility does not perform volume name checking.

This option is applicable to the backup file for which e or m is specified in the `-k` option. To specify multiple volume names, you need to specify m in the `-k` option. Make sure that each specified volume name is unique among all volumes used.

When re-creating the log point information file, specify only the first volume, even if multiple volumes are to be used (the utility references only the first volume, even if multiple volumes are specified).

(13) -N EasyMT-filename

~ <alphanumerics> ((1-17 characters))

Specifies the file name assigned when the backup file is created. This specification takes effect only on the backup file made with e or m specified in the `-k` option. An error results if the specified value does not match the input backup file.

The backup file must be created from the beginning of the mounted magnetic tape (beginning at file number 1).

(14) -G barlist-filename

～ <alphanumerics (excluding an underline (_))> <<1-64 characters>>

Specifies a barlist file name for JP1/OmniBack II that was used during the execution of the database copy utility. The barlist file must be at the server machine where JP1/OmniBack II's cell server is located.

The directory containing the barlist file is shown as follows. Note that the protection specification in the barlist file is ignored.

`/etc/opt/omni/barlists/stream`

(15) -U {backup-search-condition-start-time,backup-search-condition-end-time|,backup-search-condition-end-time}

Specifies that the database is to be recovered from a backup made at the specified time from among all the backups made with the same policy name. This option is applicable when the NetBackup linkage facility is used (`-k n` is specified). Otherwise, the utility ignores this option, if specified.

If both a backup search condition start time and a backup search condition end time are specified, the utility uses the most recent backup within the specified period for recovery. If only a backup search condition end time is specified, the utility uses the backup in effect at the specified end time for recovery.

Rules

- When this option is omitted, the utility uses the most recent backup from among the backups made with the same policy name.
- You can use the `bpimagelist` command (with the `-policy` option specified) to check the dates and times at which backups were made.
- For the backup search condition start time and backup search condition end time, connect the date and time by an underscore (_), shown below. If the time is omitted, the utility assumes `000000` for the start time and `235959` for the end time.

```
-U YYYYMMDD[_hhmmss],YYYYMMDD[_hhmmss]
```

YYYY: Year ～ <unsigned integer> ((1990-2037))

Specifies the calendar year.

MM: Month ～ <unsigned integer> ((01-12))

DD: Date ～ <unsigned integer> ((01-31)) *hh*: Hour ~ <unsigned integer> ((00-23))

mm: Minute ~ <unsigned integer> ((00-59))

ss: Second ~ <unsigned integer> ((00-59))

To specify only the end time, specify a comma immediately followed by the desired end time.

(16) -g differential-backup-group-name

~ <alphanumerics> ((1-30))

If you are using the differential backup facility, use this option to specify the differential backup group name used with `pdcopy`.

(17) -K name-of-HiRDB-file-system-area-storing-differential-backup-management-file

~ <pathname>

If you are using the differential backup facility, use this option to specify the name of the HiRDB file system area that contains the differential backup management file used with `pdcopy`.

Make sure that "*name-of-HiRDB-file-system-area-storing-differential-backup-management-file*/*differential-backup-group-name*" does not exceed 167 characters.

(18) -q generation-number

~ <unsigned integer> ((0-10))

Specifies the generation number of the RDAREA to be restored when the inner replica facility is used. When 0 is specified, the utility restores the original RDAREA; when a value in the range 1-10 is specified, the utility restores the specified replica RDAREA generation.

Rules

- Specify this option together with the `-r` option. This option cannot be specified together with the `-a`, `-c`, `-u`, or `-s` option.
- Specify this option to restore the replica RDAREA of the indicated generation number in the inner replica group to which the original RDAREA specified in the `-r` option belongs.
- The following table describes the `-q` option specification during the backup and recovery operations:

During recovery operation	During backup operation	
	-q option omitted	-q option specified
`-q` option omitted	The utility restores the RDAREAs for which a backup is available. The utility does not recover an RDAREA whose backup is not available.	
`-q` option specified	The utility restores the RDAREAs whose backup is available. The utility does not recover an RDAREA whose backup is not available.	The utility restores the RDAREAs even if their backups are not available as long as a backup of RDAREAs in the same inner replica group is available.

(19) -x generation-number

～ <unsigned integer or alphanumerics> ((0-10 or `c`))

Specifies the generation number of the RDAREA to be restored if the inner replica facility is used and the target RDAREA has been updated over multiple generations. If you specify this option, also specify the `-q` option.

`0`

Specifies that the original RDAREA is to be restored.

`1` to `10`

Specifies that the specified RDAREA generation is to be restored. For example, if you are restoring the first generation of an RDAREA using the log information for the first and second generations of the RDAREA, specify `1` in the `-q` option and `2` in the `-x` option.

`c`

Specifies that the current RDAREA in the system log file is to be restored. In this case, the utility uses log information for the current RDAREA in the system log file, ignoring the generation number.

Rules

- If no RDAREA is defined that has the generation number specified in this option, the utility terminates with an error.
- Specifying the same generation number in both the `-q` and the `-x` options will not result in an error.
- By specifying `c`, you can always restore the current RDAREA using its log information even if the RDAREA is updated over three or more generations, or multiple generations are updated at the same time. When swapping occurs on the current RDAREA, the RDAREA must have the same contents immediately before and after swapping (not only the data but also the locations within file).

- When `c` is specified, the utility supports only the log information that has been output by HiRDB version 07-00 or later. If you are upgrading your HiRDB to version 07-00, make sure that all RDAREAs have been backed up using `pdcopy` with `-a` specified.

(20) -Y write-buffer-size

~ <unsigned integer> ((4-131072))

Specifies the buffer size (in kilobytes) when recovery data is written during database recovery processing. Specifying this option reduces the number of write operations because the amount of data written at one time increases.

Criteria

Hitachi recommends that you specify this option when the following conditions are satisfied, to improve performance.

Data to be used for recovery	Condition
Differential backup file	The only data update operation performed after the previous differential backup was addition (`INSERT` statement) or deletion of all data (`DELETE` or `PURGE TABLE` statement).
Unload log file or system log file	The principal update operation performed by the database updating application is addition (`INSERT` statement).

Specification guidelines

You should specify a value in the range from 1024 to 2048.

As a guideline for the value, specify a common multiple of the page length of all RDAREAs that are subject to database recovery processing.

Rules

1. When this option is omitted, the following value is assumed:
   - When the recovery processing uses a differential backup file, unload log file, or system log file, the page length of each RDAREA that is to be recovered
   - Otherwise, 60
2. If the specified value is less than the maximum page length of the RDAREA to be recovered, the specified value is ignored and the default value is used.

(21) -f control-statement-file-name

~ <pathname>

Specifies the name of the control information file.

Example

```
-f /usr/pdrstr/cont01
```

You can specify options (3) through (5) and (22) through (27) in the control statement file. You can also specify these options directly in the `pdrstr` command.

The format of the control statement file is as follows:

```
  {{-l flag-argument|-L|-d flag-argument}|{-a|-c|-u flag-argument|-s flag-argument|-r
flag-argument}[-T flag-argument]}
  [{{-l flag-argument|-L|-d flag-argument}|{-u flag-argument|-s flag-argument|-r
flag-argument}[-T flag-argument]}]
  [{{-l flag-argument|-L|-d flag-argument}|{-u flag-argument|-s flag-argument|-r
flag-argument}[-T flag-argument]}]
                                        .
                                        .
                                        .
```

Notes

- Create the control statement file at the server machine where the database recovery utility was executed (where the `pdrstr` command was entered).
- In the control statement file, you must specify the combination of target RDAREA specification options and the `-T` option on a single line.
- A line in the control statement file must not exceed 32,768 bytes.
- For details about the option flags, see the explanations of the individual options.
- If the `-u`, `-s`, or `-r` option consists of multiple lines, you can specify on the subsequent lines only those option flags specified on the first line (except for the `-a` and `-c` options). For example, if you specify the `-u` option on the first line, you can specify the `-u` option on any subsequent line.
- If the `-l` option is specified on multiple lines, the last specification takes effect; the same applies to the `-d` option.
- An option specified on the command line cannot also be specified in the control statement file. However, the `-T` option can be specified on both the command line and in the control statement file, in which case the `-T` option specification on the command line applies to any item for which the `-T` option is omitted in the control statement file.
- Specification of a control statement file may not be supported depending on the combination of options specified on the command line. The following table shows the combinations of options and whether or not a control statement file can be specified:

-l, -L, or -d option	**-a, -c, -u, -s, or -r option**	
	Omitted	**Specified**
Omitted	Y[1]	Y[2]
Specified	Y[3]	N

Legend:

Y: Can be specified

N: Cannot be specified

[1] All options specifiable in the control statement file can be specified.

[2] The `-l`, `-L`, or `-d` option can be specified.

[3] The `-a`, `-c`, `-u`, `-s`, or `-r` option and the `-T` option can be specified.

Example of correct control statement

```
-r pdbuser1 -T 20040809,20040810
-r pdbuser2 -T 20040809
-l /unldlog/unld01,/unldlog/unld02,
/unldlog/unld03,/unldlog/unld04
```

This example uses the database update history stored in the unload log files `unld01`, `unld02`, `unld03`, and `unld04` to recover the database. To recover the RDAREA `pdbuser1`, the example uses the database update history from 2004-08-09 00:00:00 to 2004-08-10 24:59:59. To recover the RDAREA `pdbuser2`, the example uses the database update information starting at 2004-08-09 00:00:00.

Example of incorrect control statement (1)

```
-r pdbuser1 -T 20040809,20040810
/unldlog/unld01,/unldlog/unld02     ← Error
```

A line containing the `-l`, `-L`, or `-d` option cannot contain any other option.

Example of incorrect control statement (2)

```
-r pdbuser1 -T 20040809,20040810
-r pdbuser2 -T 20040809
-l /unldlog/unld01
,/unldlog/unld02     ← Error
```

Because there is a limit to the length of a single line (maximum of 32,768 bytes including linefeed code), when the `-l` or `-d` option has a long argument, you can enter a linefeed code and continue specification onto the next line. The linefeed code can be entered only after the comma (`,`) that separates the path name of an unload log file. If the line specifying the `-l` or `-d` option ends with a character other than the comma, the utility treats it as the end of the line.

(22) -a

Specifies that all RDAREAs are to be restored (except for list RDAREAs).

Rule

- If you specify the `-a` option, the control statement file can contain only one line of control statement.
- If a replica RDAREA has been defined, it will also be subject to recovery processing.

(23) -c

Specifies that all RDAREAs in the backup files specified with the `-b` option (except the list RDAREAs) are to be restored.

Rules

- If the `-c` option is specified, the control statement file can contain only one line of control statement.
- If you are specifying the `-c` option, be sure to also specify the `-b` option. However, when using the differential backup facility, you do not need to specify the `-b` option (if the `-b` option is specified, the utility assumes manual recovery; if the `-b` option is omitted, the utility assumes automatic recovery).
- If you are specifying the `-c` option, do not specify the `-l` option.
- If the backup file contains a backup of a replica RDAREA, the utility restores the source generation of that replica RDAREA.
- If you made a backup copy with the `pdcopy` command with the `-J` option specified and the `KFPR26063-I` was issued, recovery processing using that backup copy will not result in normal termination (because the skipped

RDAREAs are also subject to recovery processing).

(24) -u unit-identifier[,unit-identifier]...

~ <identifier> ((4 characters))

If all RDAREAs under specified units are to be restored (except the list RDAREAs), use this option to specify the unit identifiers.

Rules

- If specifying multiple unit identifiers, make sure none of them is duplicated.
- When using a control statement file, make sure that each specified unit identifier is unique among all the control statements.
- If a replica RDAREA has been defined, it will also be subject to recovery processing.

(25) -s server-name[,server-name]...

~ <identifier> ((1-8 characters))

If all RDAREAs under specified servers are to be restored (except the list RDAREAs), use this option to specify the server names.

Rules

- If specifying multiple server names, make sure none of them is duplicated.
- When using a control statement file, make sure that each specified server name is unique among all the control statements.
- If a replica RDAREA has been defined, it will also be subject to recovery processing.

(26) -r RDAREA-name[,RDAREA-name]...

~ <identifier> ((1-30 characters))

Specifies the names of the RDAREAs to be restored.

Rules

- You cannot specify a list RDAREA.
- If specifying multiple RDAREA names, make sure none of them is duplicated. When using a control statement file, make sure that each specified RDAREA name is unique among all the control statements (except for a regular expression).
- You can specify RDAREA names in regular expressions. For details, see the explanation of the `-r` option in Chapter *18. Database Copy Utility (pdcopy)*.
- The following rules apply to the RDAREA names specified in the `-r` option:

Type	Coding method without using regular expression	Coding method using regular expression
Command line specification (without using a control statement file)	When specifying an RDAREA name in lowercase alphabetic characters or when the RDAREA name contains a space, enclose the entire name in double quotation marks (`"`). If you use the Bourne shell (sh), C shell (csh), or Korn shell (ksh), you also need to enclose the RDAREA name in single quotation marks (`'`).	Enclose the entire regular expression with single quotation marks (`'`). Regular expression is case sensitive.
Specifying in a control statement file	When specifying an RDAREA name in lowercase alphabetic characters or when the RDAREA name contains a space, enclose the entire name in double quotation marks (`"`).	Regular expression is case sensitive.

- If the `-q` and `-r` options are both specified, only system RDAREAs and original RDAREAs can be specified. In this case, in the inner replica group to which the original RDAREA belongs, the replica RDAREA of the generation specified in the `-q` option is subject to processing.
- If the replica RDAREA with the generation specified in the `-q` option is undefined, the utility ignores this RDAREA and restores another RDAREA.
- When the `-q` option is specified, the regular expression is applicable to the original RDAREA. If the `-q` option is omitted, the regular expression is applicable to all RDAREAs including the replica RDAREAs.

(27) -T{recovery-start-time,recovery-end-time|recovery-start-time |recovery-end-time}

Use this option to specify a recovery range.

When the `-l`, `-d`, or `-L` option is specified, specifies the output times of the first and last log information items to be used for database recovery.

Rules

- If you specify the `-l` or `-L` option but omit this option, the utility assumes the output times of the first and last system log files or unload log files.
- If you omit the `-l` or `-L` option, the utility ignores this option, if specified, unless the differential backup facility is used.
- The recovery start time must be earlier than the recovery end time. If the specified recovery start time is later than the recovery end time, an error results and processing terminates. When you are specifying this option, make sure that a recovery start time, a recovery end time, or both are specified.

- The recovery start time and recovery end time are expressed as a date and time separated by an underscore (_), as shown below. If the time is omitted, the utility assumes `000000` for the recovery start time and `235959` for the recovery end time.

```
-T YYYYMMDD[_hhmmss],YYYYMMDD[_hhmmss]
```

YYYY: year ~ <unsigned integer> ((1990-2037))

MM: month ~ <unsigned integer> ((01-12))

DD: Date ~ <unsigned integer> ((01-31))

hh: hour ~ <unsigned integer> ((00-23))

mm: minute ~ <unsigned integer> ((00-59))

ss: second ~ <unsigned integer> ((00-59))

If the recovery end time only is specified, it must be preceded by a comma (`,`); if the recovery start time only is specified, it must not be followed by a comma.

Notes

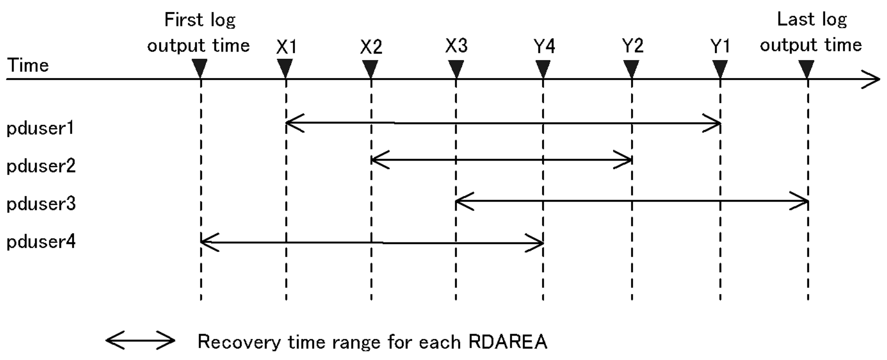
1. When executing recovery with a time range specified, be sure to specify the time of the server machine that contains the RDAREAs being recovered because the time differs from one server to another.
2. When using a backup file to execute recovery with a time range specified, there is no need to specify the recovery start time. The utility ignores the recovery start time, if specified, and assumes the backup start time. For the recovery end time, specify a time that falls after the backup end time.
3. When restoring multiple individual RDAREAs updated by a single transaction with a time range specified, specify a time that falls after the transaction end time (synchronization point) for the recovery end time. An error results if the specified time falls before the transaction end time, in which case you need to specify the correct time (which falls after the transaction end time) again.
4. The following shows an example of the `pdrstr` command specifying the `-T` option and the control statement file specifying the `-T` option in the `-r` option:

Command specification

```
pdrstr ...
    -T X1,Y1
    ...
    -f control-statement-file-1
```

Contents of control statement file 1

```
-r pduser1
-r pduser2 -T X2,Y2
-r pduser3 -T X3
-r pduser4 -T ,Y4
```



Note:
When the recovery start time or the recovery end time is omitted in the control statement file, the recovery start time or recovery end time specified by the command is not inherited. Specified values are inherited only when both the recovery start time and the recovery end time are omitted (which is the case for `pduser1`).

(28) -z log-point-information-file-name

~ <pathname>

When re-creating the log point information file, use this option to specify the name of the file. This file is created at the host where the `pdrstr` command is entered.

Rules

- If specifying this option, you need to specify the backup file name with the `-b` option. This backup file must have been created by the `pdcopy` command specifying the `-z` option.
- For the log point information file name, specify an absolute path name

beginning with the root (/). If the specified file already exists, the utility deletes this file before re-creating the log point information file.

19.5 Notes

1. A maximum of 1120 RDAREAs can be restored per server by a single execution of the database recovery utility.
2. During database recovery, no other user can access an RDAREA subject to recovery processing.
3. When executing the database recovery utility, do not specify an unload log file that was created before executing the database initialization utility. If such an unload log file is specified, the operation cannot be guaranteed.
4. To restore an extracted database subject to data linkage, use the backup copy of the database and the log obtained after the backup was made. If you use any other method to restore the database, inconsistency results between the extracted database and the target database.
5. If the database recovery utility execution is cancelled due to a power failure or by the `kill` command, the HiRDB file system area may be damaged. If the `pdfls` command detects inconsistency, restore the HiRDB file system area using one of the following methods:
   - Re-create the HiRDB file system area using the `pdfmkfs` command and execute the recovery utility for each RDAREA to restore the HiRDB files contained in the HiRDB file system area.
   - Execute the `pdfbkup` and `pdfrstr` commands to back up and restore the HiRDB file system area.
6. If a regular file is used for the HiRDB file system area that contains the RDAREAs subject to recovery processing, and the `pdfmkfs` command is executed without specifying the `-I` option, you need to use the backup file created by the database copy utility to restore database from the log.
7. If both EasyMT and JP1/Magnetic Tape Access are installed in the system, the utility starts JP1/Magnetic Tape Access.
8. The maximum number of copies of the database recovery utility that can be executed concurrently depends on the value of the `pd_utl_exec_mode` operand in the system common definition:
   - `pd_utl_exec_mode=0`

     You can execute a maximum of 32 copies concurrently.
   - `pd_utl_exec_mode=1`

     You can execute as many copies as there are specified in the `pd_max_users` operand.

9. If you made a backup copy with the `pdcopy` command with the `-J` option specified and some RDAREAs were skipped, recovery processing using that backup copy will not restore the skipped RDAREAs.

10. The return codes of `pdrstr` are as follows:

    `0`: Normal termination

    `4`: Warning termination (recovery processing terminated normally, but an error occurred at the warning level that is unrelated to recovery processing)

    `8`: Abnormal termination (an error occurred during recovery processing, but some RDAREAs were restored)

    `12`: Abnormal termination

11. If you use Real Time SAN Replication based on the log-only synchronous method and execute `pdrstr` at the transaction execution site, you must perform the preparations for log application. For details about the preparations for log application, see the manual *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide*.

12. If you selected `utf-8` as the character encoding in the `pdsetup` command, you can use a file with a BOM as the control statements file for `pdrstr`. Note that even when a file with a BOM is used as the control statements file, the BOM is skipped. No BOM is included in the file that is output by `pdrstr`.

19.6 Database recovery utility process results listing

The following shows an example of the process results output files for the database recovery utility.

```
pdrstr(VV-RR)     ***** DB RECOVERY *****   1995-05-16 11:35:35 [1] utl3 [2]
*** DB RECOVERY INFORMATION LIST ***
---------------------------------------------------------
<<RDAREA INFORMATION>>
  UNIT NAME  :      un16 [3] SERVER NAME:sds [4]
  RDAREA NAME: pdbuser01 [5]
  RDAREA ID :          4 [6] ATTRIBUTE : USER [7]      PAGE SIZE :  4096 [8]
  RECOVERY STARTED AT : 1995-05-16 11:34:19 [9]
  RECOVERY ENDED AT   : 1995-05-16 11:34:46 [10]

 <REPLICA RDAREA INFORMATION>
  ORIGINAL RDAREA NAME : pdbuser01 [19]
  REPLICA RDAREA GENERATION NO : 0 [20]
<<BACKUP INFORMATION>>
  BACKUP STARTED AT : 1995-05-16 11:11:42 [11]
  BACKUP ENDED AT   : 1995-05-16 11:11:42 [12]
<<LOG INFORMATION>>
  1ST LOG    : 1995-05-16 10:57:48 [13]
  LAST LOG   : 1995-05-16 10:58:26 [14]
  LOG COUNT  :         42 [15]

 <FILE INFORMATION>
 1 /dbarea/area1/rdmt06 [16]
   EXTENT COUNT :   2 [18]
   PAGE NUMBER  :    124 [17]
 2 /dbarea/area1/rdmt07 [16]
   EXTENT COUNT :   2 [18]
   PAGE NUMBER  :    108 [17]
--------------------------------------------------------

<<RDAREA INFORMATION>>
  UNIT NAME  :      un16 SERVER NAME:sds
  RDAREA NAME: pdbuser03
  RDAREA ID :           5 ATTRIBUTE : USER            PAGE SIZE :  4096
  RECOVERY STARTED AT : 1995-05-16 11:34:46
  RECOVERY ENDED AT   : 1995-05-16 11:35:10

 <REPLICA RDAREA INFORMATION>
  ORIGINAL RDAREA NAME : pdbuser03
  REPLICA RDAREA GENERATION NO : 0
<<BACKUP INFORMATION>>
  BACKUP STARTED AT : 1995-05-16 11:11:42
  BACKUP ENDED AT   : 1995-05-16 11:11:43
<<LOG INFORMATION>>
  1ST LOG    : 1995-05-16 10:59:50
  LAST LOG   : 1995-05-16 11:02:35
  LOG COUNT  :         42
 <FILE INFORMATION>
 1 /dbarea/area1/rdmt08
   EXTENT COUNT :   2
   PAGE NUMBER  :    124
 2 /dbarea/area1/rdmt09
   EXTENT COUNT :   2
   PAGE NUMBER  :    108
--------------------------------------------------------

<<RDAREA INFORMATION>>
  UNIT NAME  :      un16 SERVER NAME:sds
  RDAREA NAME: pdbuser05
  RDAREA ID :           6 ATTRIBUTE : USER            PAGE SIZE :  4096
  RECOVERY STARTED AT : 1995-05-16 11:35:10
  RECOVERY ENDED AT   : 1995-05-16 11:35:34

 <REPLICA RDAREA INFORMATION>
  ORIGINAL RDAREA NAME : pdbuser05
  REPLICA RDAREA GENERATION NO : 0
<<BACKUP INFORMATION>>
  BACKUP STARTED AT : 1995-05-16 11:11:43
  BACKUP ENDED AT   : 1995-05-16 11:11:43
<<LOG INFORMATION>>
  1ST LOG    : 1995-05-16 11:03:21
  LAST LOG   : 1995-05-16 11:08:47
  LOG COUNT  :         42

 <FILE INFORMATION>
 1 /dbarea/area1/rdmt10
   EXTENT COUNT :   2
   PAGE NUMBER  :    124
 2 /dbarea/area1/rdmt11
   EXTENT COUNT :   2
   PAGE NUMBER  :    108
--------------------------------------------------------
```

Explanation

1. Date and time the database recovery utility was executed (in the format *YYYY*/*MM*/*DD* *hh*:*mm*:*ss*).

   YYYY: Year. *MM*: Month. DD: Date. *hh*: Hour. *mm*: Minute. *ss*: Second.
2. HiRDB identifier
3. Name of the unit containing an RDAREA
4. Name of the unit containing an RDAREA
5. Name of the recovered RDAREA
6. Number of the recovered RDAREA
7. Type of RDAREA:

   `MASTERDIRECTORY`: Master directory RDAREA

`DATADIRECTORY`: Data directory RDAREA

`DATADICTIONARY`: Data dictionary RDAREA

`SYSTEM_LOB`: Data dictionary LOB RDAREA

`USER`: User RDAREA

`USER_LOB`: User LOB RDAREA

`REG`: Registry RDAREA

`REG_LOB`: Registry LOB RDAREA

8. Page length of the RDAREA (in bytes)
9. Recovery start time of the RDAREA (format is the same as in 1)
10. Recovery end time of the RDAREA (format is the same as in 1)
11. Backup start time of the RDAREA (format is the same as in 1)
12. Backup end time of the RDAREA (format is the same as in 1)
13. Time the first log record was output during recovery processing (format is the same as in 1)
14. Time the last log record was output during recovery processing (format is the same as in 1)
15. Number of updated log records used during recovery processing
16. Name of a HiRDB file constituting the RDAREA
17. Number of pages allocated to the HiRDB file
18. Number of extents in the HiRDB file
19. Name of the original RDAREA
20. Generation number of the replica RDAREA

Notes

1. If there is no data to be restored, the utility displays `No data for recovery` at 9, and displays nothing at 8-15.
2. If there is no backup file, the utility does not display 11 or 12.
3. If there are no log records for an RDAREA, the utility does not display 13 or 14.
4. 19 and 20 are output if the inner replica facility is used.

Chapter

20. Registry Facility Initialization Utility (pdreginit)

This chapter explains the registry facility initialization utility (`pdreginit`), which can be used to register registry RDAREAs and registry LOB RDAREAs for managing and operating the registry facility and to create registry management tables.

This chapter contains the following sections:

20.1 Overview

Executor

HiRDB administrators can execute this command.

20.1.1 Function

The registry facility initialization utility can be used to install the registry facility in the HiRDB system so that plug-in modules can be used.

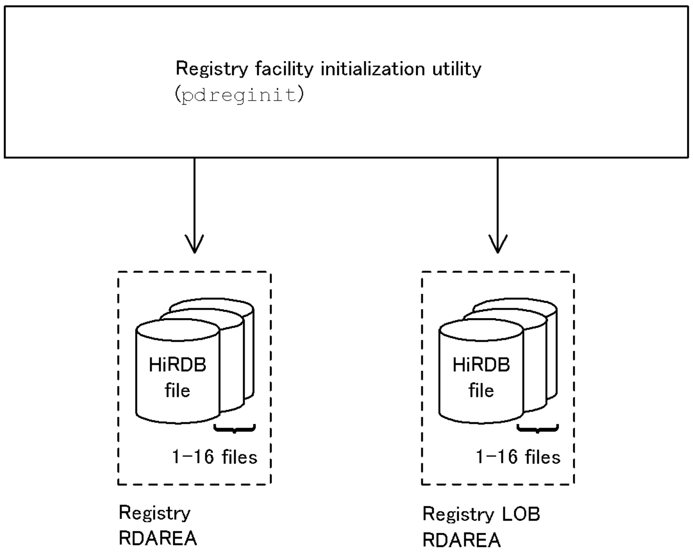
Adding a registry RDAREA and a registry LOB RDAREA using the registry facility initialization utility creates a registry management table for keeping track of the registry information used by a plug-in.

Preconditions

Before the Registry facility initialization utility can be executed, the HiRDB system must be enabled to use the stored procedure function. Whether the stored procedure function is enabled can be tested by using the `pddbls` command. Use the pddbls command whether a data dictionary LOB RDAREA is available. The stored procedure function can be used if a data dictionary LOB RDAREA is available. If a data dictionary LOB RDAREA does not exist, add a data dictionary LOB RDAREA by using the database structure modification utility to enable the stored procedure function.

Figure 20-1 shows an overview of the registry facility initialization utility (`pdreginit`).

Figure 20-1: Overview of the registry facility initialization utility (pdreginit)



20.2 Command format

20.2.1 Format

```
pdreginit [-k processing-type][-a control-statement-file-name]
```

20.2.2 Options

-k *processing-type*

Specifies whether a registry RDAREA or a registry LOB RDAREA is to be added, or only a registry manipulation stored procedure is to be registered. The default is all.

`all`

Specifies that any of the following operations are to be performed: a control statement file must first be specified in the `-a` option:

- Add a registry RDAREA and a registry LOB RDAREA.
- Create a registry management table for using the registry facility.
- Register a stored procedure for operating the registry.

`proc`

Specifies that a registry operation stored procedure only is to be registered.

If the registry facility initialization utility terminates with an error after normal termination of the addition of a registry RDAREA (the database structure modification utility produced a `KFPX24200-I` message with a return code of `0` or `4`), this option can be specified to register the registry operation stored procedure only. The `-a` option cannot be specified together with this option.

`renew`

Specifies that a registry operation stored procedure is to be reregistered.

This option can be specified to re-register a registry operation stored procedure when moving from a HiRDB/Single Server to HiRDB/Parallel Servers. The `-a` option cannot be specified together with this option.

-a *control-statement-file-name* ~ <path-name> ((up to 255 characters))

On HiRDB/Parallel Servers, the control statement file needs to be referenced from the node at which the dictionary server is located.

In this control statement, the `create rdarea` statement must be specified for each of the registry RDAREA and the registry LOB RDAREA. If a registry RDAREA or a registry LOB RDAREA already exists, specification of this option causes an error.

20.3 Control statements

20.3.1 create rdarea statement

This section explains the operands of the `create rdarea` statement. In the following table, each number corresponds to the number assigned to each operand.

No.	Operand
1	`create rdarea` *RDAREA-name*
2	`[globalbuffer` *global-buffer-name*`]`
3	`for {registry|LOB used by HiRDB (SQL REGISTRY)}`
4	`[page` *page-length* `characters]`
5	`[storage control segment` *segment-size* `pages]`
6	`[extension {use` *extension-segments-count* `segments|`<u>`nouse`</u>`}]`
7	`file name "`*HiRDB file-system-area-name/HiRDB-file-name*`"`
8	`initial` *HiRDB-file-segments-count* `segments`
—	`[file name "`*HiRDB file-system-area-name/HiRDB-file-name*`"`
—	`initial` *HiRDB-file-segments-count* `segments]...`
—	`;`

(1) RDAREA-name

~ <identifier> ((1-30))

Specify the name of the RDAREA to be used in the HiRDB system. `ALL` cannot be specified in an RDAREA name. When enclosed in double quotation marks ("), an RDAREA name is case sensitive; otherwise, it is not treated as being all in uppercase.

Examples

- `create rdarea "pdbuser01" for ...`

  In this case, the RDAREA name is treated as `pdbuser01`.
- `create rdarea pdbuser01 for ...`

  In this case, the RDAREA name is treated as `PDBUSER01`.

(2) globalbuffer global-buffer-name

~ <identifier> ((1-16))

Specifies the name of a global buffer in the HiRDB system (or in the dictionary server for a HiRDB/Parallel Server) to temporarily allocate it to the RDAREA to be added. You cannot specify an index global buffer. To check the global buffer, use the `pdbufls` command.

Rules

1. This operand is required for the addition of a registry RDAREA.
2. The specified global buffer must have a buffer length greater than or equal to the page length of the RDAREA to be added. To check the size of global buffer, use the `pdbufls` command.
3. The global buffer length is the value specified in the `-l` operand of the `pdbuffer` statement in the system common definitions, or the maximum page length for the RDAREA to which the global buffer is assigned at the time of HiRDB startup, whichever is greater.
4. The specification of this operand loses effect once the HiRDB system starts normally. Therefore, you need to change the global buffer specification in the system common definitions at the next normal startup.
5. If you specify this operand, but the utility cannot allocate a global buffer, then an RDAREA cannot be added.

(3) for {registry |LOB used by HiRDB (SQL_REGISTRY)}

Specifies the type of RDAREA to be added.

`registry`

This option is specified if the RDAREA to be added is a registry RDAREA.

`LOB used by HiRDB (SQL_REGISTRY)`

This option is specified if the RDAREA to be added is a registry LOB RDAREA.

(4) page page-length characters

~ <unsigned integer> ((4096-30720)) <<4096 or 8192>>

Specifies the page length of the HiRDB file comprising the RDAREA in bytes, in an integral multiple of 2048.

Registry LOB RDAREA

A value of 8192 must be specified. Even if other values are specified, 8192 is assumed.

The page length specified in this operand is used as the smallest unit of data that is input/output by the HiRDB system to or from the RDAREA.

(5) storage control segment segment-size pages

~ <unsigned integer> ((1-16000)) <<1 or 50>>

Specifies the size of a segment in terms of pages.

Registry LOB RDAREA

A value of 1 must be specified. Even if other values are specified, 1 is assumed.

(6) extension {use extension-segments-count segments|nouse}

Specifies whether or not to apply automatic extension of RDAREA.

In the event of a space shortage in an RDAREA, the RDAREA automatic extension facility expands the RDAREA automatically if there is enough space in the HiRDB file system area. If you apply this facility to an RDAREA and a shortage of unused segments occurs, the system allocates new unused segments to the RDAREA. These new unused segments are added at the end of the HiRDB file constituting the RDAREA.

Prerequisites

1. You need to specify the `-e` option (specifying the number of extensions) for the HiRDB file system area containing the RDAREA.
2. There must be enough space in the HiRDB file system area that contains the last HiRDB file constituting the RDAREA.

`use` *extension-segments-count* `segments`

Specifies that automatic extension of RDAREA is to be applied.

In the case of a registry RDAREA, automatic extension occurs when there are no more used free segments or used segments in the RDAREA. In the case of a registry LOB RDAREA, automatic extension occurs when there are no more unused segments.

extension-segments-count ~ <unsigned integer> ((1-64000))

Specifies the number of extension segments.

The maximum number of HiRDB file extensions is 24. If this value is exceeded, an error occurs. The maximum number of extensions per HiRDB file system area is determined by the value specified when the HiRDB file system area is created. Therefore, you need to define the maximum number of extensions, taking into account the number of files in the HiRDB file system area and the frequency of extension.

`nouse`

Specifies that automatic extension of RDAREA is not to be applied.

Notes

1. If allocation of unused segments fails due to a shortage of space in the HiRDB file system area, either you must extend or re-initialize the RDAREA or you must use the database reorganization utility to reorganize the data dictionary table.
2. If the number of extents exceeds the maximum value, integrate the extents in the HiRDB file system area containing the RDAREA or add HiRDB file in another HiRDB file system area to the RDAREA.

   To integrate extents, make a backup copy with `pdfbkup`, initialize the HiRDB file system area with `pdfmkfs`, then restore the HiRDB file system area from its backup copy using `pdfrstr`.
3. The last file is locked from the beginning to the end of the automatic extension process.

(7) filename-"HiRDB-file-system-area-name/HiRDB-file-name"

~ ((up to 167 characters))

Specifies the name of the HiRDB file system area and the name of the HiRDB file to be allocated to the RDAREA.

Rules

1. Enclose *HiRDB-file-system-area-name*/*HiRDB-file-name* in double quotation marks (`"`).
2. Do not include a linefeed character inside the double quotation marks.
3. You can allocate a maximum of 16 HiRDB files per RDAREA.
4. *HiRDB-file-system-area-name*/*HiRDB-file-name* must be unique in the HiRDB system.

HiRDB-file-system-area-name ~ <pathname>

Specifies the name of the HiRDB file system area.

HiRDB-file-name ~ < HiRDB filename> (1-30 characters))

Specifies the name of a HiRDB file not beginning with `pl`.

(8) initial HiRDB-file-segments-count segments

~ <unsigned integer>

Specifies the number of segments for the HiRDB file, such that the size of the HiRDB file will not exceed 64 GB.

20.3.2 Notes

1. Make sure that the number of RDAREAs specified does not exceed the maximum number of RDAREAs specified in the system common definition

(`pd_max_rdarea_no`). Similarly, make sure that the number of HiRDB files constituting the RDAREAs does not exceed the maximum number of HiRDB files that can constitute RDAREAs (`pd_max_file_no`). If it is necessary to make a specification that exceeds one of these limits, you must use the `pdchgconf` command to change the system common definition or terminate the HiRDB system normally and then change the system common definition. For details about the system common definition, see the manual *HiRDB Version 8 System Definition*.

2. A comment can be added in a control statement by enclosing it in a slash-asterisk (`/*`) asterisk-slash (`*/`) combination.

20.4 Rules and notes

(1) Rules

1. The registry facility initialization utility can be executed only when HiRDB is running.
2. The registry facility initialization utility should be executed on either a single-server or a server machine with a system manager.
3. The registry facility becomes available after the registry facility initialization utility has terminated and a global buffer is allocated to the registry LOB RDAREA that has been added. Registry information for plug-ins can be registered after the registry facility has become available.
4. Before executing the registry facility initialization utility, define a HiRDB file system area on a single-server or on a dictionary server. The `pdfmkfs` command can be used to initialize the HiRDB file system area.
5. When executing the registry facility initialization utility, set appropriate client environment variables (`PDNAMEPORT`, `PDHOST`, and `PDUSER`) and an authorization identifier with `CONNECT` privileges. For details about the client environment variables, see the *HiRDB Version 8 UAP Development Guide*.
6. After completing the execution of the registry facility initialization utility, be sure to use the database copy utility (`pdcopy`) to back up the registry RDAREAs, the registry LOB RDAREAs, the master directory, and the data dictionary LOB RDAREAs that were added.

(2) Notes

1. The following are the `pdreginit` command's return codes:

   `0`: Normal termination

   `4`: Normal termination (warning-level error occurred, but processing terminated normally)

   `8`: Abnormal termination
2. The addition of registry RDAREA and registry LOB RDAREAs and the creation of a registry management table are performed when control is passed to the database structure modification utility (`pdmod`).
3. The registration of a registry operation stored procedure is executed upon normal termination of the database structure modification utility (with a `KFPX24200-I` message with a return code of `0` or `4`). In the event of abnormal termination of the database structure modification utility (with a `KFPX24200-I` message with a return code of `12`), the addition of the RDAREA and the creation of the registry

management table will be rolled back.

4. If an error occurred during the registration of a registry operation stored procedure, specify the registration of the registry operation stored procedure only, using the `-k` option, when executing the registry facility initialization utility next time. An attempt to specify the addition of a registry RDAREA and a registry LOB RDAREA without registering a registry operation stored procedure may cause an error.
5. If you selected `utf-8` as the character encoding in the `pdsetup` command, you can use a control statements file that contains a BOM. However, only ASCII characters are permitted for comments in the control statements file. If character encoding other than ASCII is used, `pdreginit` may not function correctly.

20.5 Examples

Example 1 shows how to use the registry facility initialization utility.

Example 1

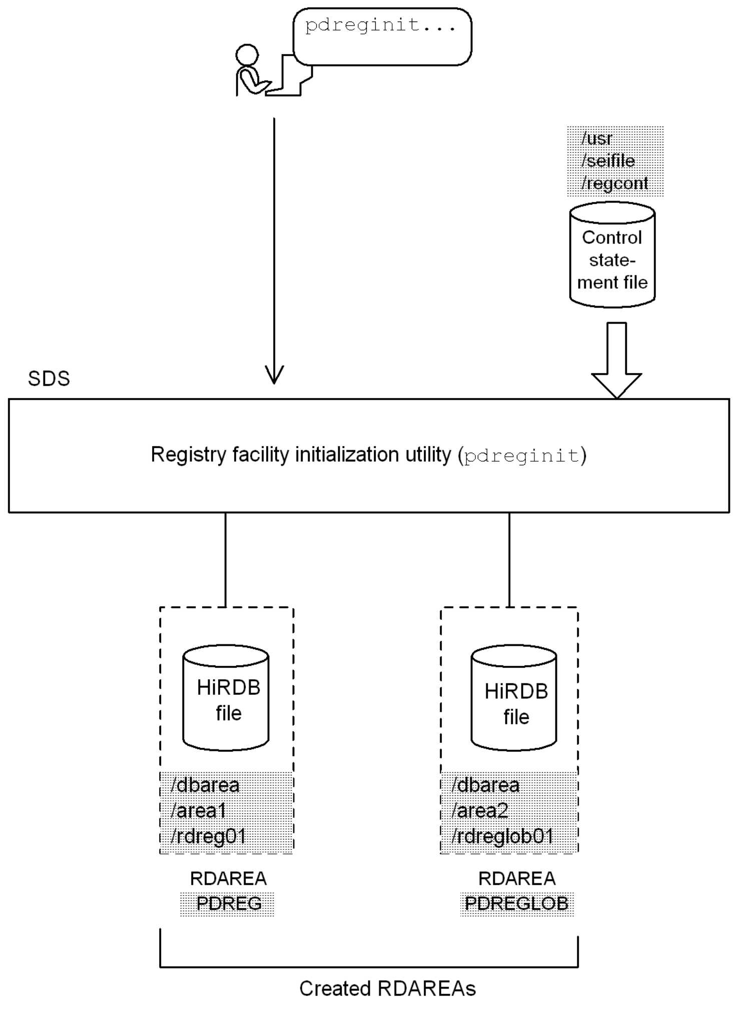
Create the following RDAREAs:

- `PDREG` (registry RDAREA)
- `PDREGLOB` (registry LOB RDAREA)

Assume that the following HiRDB file system areas are already created:

- `/dbarea/area1` (regular file)
- `/dbarea/area2` (regular file)

Overview



▨: Value specified in the registry facility initialization utility
SDS: Single server

Example of command execution

```
pdreginit -a /usr/seifile/regcont
```

Contents of the control statement file (/usr/seifile/regcont)

```
/* Defining registry RDAREA */
 create rdarea PDREG globalbuffer gb1   ..................1
              for registry   ............................2
              page 4096 characters   ....................3
              storage control segment 50 pages   ........4
              file name "/dbarea/area1/rdreg01"
                  initial 50 segments ;   ...............5


/* Defining registry LOB RDAREA */
 create rdarea PDREGLOB globalbuffer gb2   ...............6
              for LOB used by HiRDB(SQL_REGISTRY)   .....7
              page 8192 characters   ....................8
              storage control segment 1 pages   .........9
              file name "/dbarea/area2/rdreglob01"
                  initial 1000 segments ;   .............10
```

Explanation

1. Name of the RDAREA: `PDREG`
2. Type of the RDAREA: registry RDAREA (stores registry management information)
3. Page length: 4096 characters
4. Segment size: 50 pages
5. Specification of the HiRDB file comprising the RDAREA:

   Name: `/dbarea/area1/rdreg01`

   Number of segments: 50
6. Name of the RDAREA: `PDREGLOB`
7. Type of RDAREA: registry LOB RDAREA
8. Page length: 8192 characters
9. Segment size: 1 page
10. Specification of the HiRDB file comprising the RDAREA:

    Name: `/dbarea/area2/rdreglob01`

    Number of segments: 1000

Appendixes

A. List of Commands

HiRDB provides operation commands and utilities. Table A-1 lists the commands that can be used with HiRDB.

Table A-1: List of commands

Type	Command	Description	Reference
System operation	`pdadmvr`	Displays the HiRDB version information	`pdadmvr`
	`pdcat`	Displays file contents	`pdcat`
	`pdchgconf`	Changes the system configuration	`pdchgconf`
	`pdconfchk`	Checks system definitions	`pdconfchk`
	`pdcspool`	Deletes troubleshooting information and temporary work files	`pdcspool`
	`pdgen`	System generator	`pdgen`
	`pdgeter`	Collects error information	`pdgeter`
	`pditvstop`	Stops periodic acquisition of the HiRDB status	`pditvstop`
	`pditvtrc`	Acquires the HiRDB status periodically	`pditvtrc`
	`pdjarsync`	Manipulates JAR files	`pdjarsync`
	`pdlistls`	Displays list definition information	`pdlistls`
	`pdlodsv`	Reduces the size of the installation directory	`pdlodsv`
	`pdls`	Displays the HiRDB system status	`pdls`
	`pdmemsv`	Saves memory space	`pdmemsv`
	`pdobjconv`	Migrates SQL objects into a 64-bit-mode HiRDB	`pdobjconv`
	`pdopsetup`	Sets up optional program products	`pdopsetup`
	`pdsetup`	Registers or deletes a HiRDB system in the OS	`pdsetup`
	`pdsvhostname`	Displays the server's host name	`pdsvhostname`
	`pdvrup`	Upgrades HiRDB	`pdvrup`
HiRDB file system	`pdfbkup`	Backs up the HiRDB file system	`pdfbkup`

Type	Command	Description	Reference
	`pdffsck`	Checks and repairs the integrity of a HiRDB file system area	`pdffsck`
	`pdfls`	Displays HiRDB file system information	`pdfls`
	`pdfmkfs`	Initializes a HiRDB file system area	`pdfmkfs`
	`pdfrm`	Deletes a HiRDB file	`pdfrm`
	`pdfrstr`	Restores the HiRDB file system	`pdfrstr`
	`pdfstatfs`	Displays the status of a HiRDB file system area	`pdfstatfs`
Log files	`pdlogadpf`	Allocates a log-related file	`pdlogadpf`
	`pdlogatul`	Manages the automatic log unloading facility	`pdlogatul`
	`pdlogchg`	Changes the status of a log-related file	`pdlogchg`
	`pdlogcls`	Closes a log-related file	`pdlogcls`
	`pdloginit`	Initializes a log-related file	`pdloginit`
	`pdlogls`	Displays log-related file information	`pdlogls`
	`pdlogopen`	Opens a log-related file	`pdlogopen`
	`pdlogrm`	Deletes a log-related file	`pdlogrm`
	`pdlogswap`	Swaps log-related files	`pdlogswap`
	`pdlogsync`	Collects a synchronization point dump	`pdlogsync`
	`pdlogucat`	Displays information about unload log files	`pdlogucat`
	`pdlogunld`	Unloads a log-related file	`pdlogunld`
Status files	`pdstscls`	Closes a status file	`pdstscls`
	`pdstsinit`	Initializes a status file	`pdstsinit`
	`pdstsopen`	Opens a status file	`pdstsopen`
	`pdstsrm`	Deletes a status file	`pdstsrm`
	`pdstsswap`	Swaps status files	`pdstsswap`
HiRDB startup and termination	`pdstart`	Starts a HiRDB system, unit, or server	`pdstart`
	`pdstop`	Terminates a HiRDB system, unit, or server	`pdstop`
Statistics log	`pdstbegin`	Starts output of statistical information	`pdstbegin`

Type	Command	Description	Reference
	`pdstend`	Stops output of statistical information	`pdstend`
	`pdstjswap`	Swaps statistics log files	`pdstjswap`
	`pdstjsync`	Applies the statistics log buffer to the statistics log file	`pdstjsync`
RDAREAs	`pdclose`	Closes RDAREAs	`pdclose`
	`pddbls`	Displays the status of RDAREAs	`pddbls`
	`pdhold`	Shuts down RDAREAs	`pdhold`
	`pdopen`	Opens RDAREAs	`pdopen`
	`pdrels`	Releases RDAREAs from shutdown status	`pdrels`
	`pddbfrz`	Freezes updating of a full HiRDB file constituting a user LOB RDAREA	`pddbfrz`
	`pdrdrefls`	Displays information about related RDAREAs	`pdrdrefls`
Global buffer	`pdbufls`	Displays global buffer information	`pdbufls`
	`pdbufmod`	Updates a global buffer dynamically	`pdbufmod`
Transaction control	`pdcmt`	Commits a transaction	`pdcmt`
	`pdfgt`	Terminates a transaction forcibly	`pdfgt`
	`pdrbk`	Rolls back a transaction	`pdrbk`
	`pdtrndec`	Settles an unsettled transaction forcibly and automatically	`pdtrndec`
Process control	`pdcancel`	Terminates a UAP or utility forcibly	`pdcancel`
	`pdchprc`	Changes the number of resident server processes	`pdchprc`
	`pdpfresh`	Refreshes a server process	`pdpfresh`
	`pdrpause`	Restarts a process service	`pdrpause`
Updating HiRDB version	`pdprgcopy`	Copies a HiRDB update version	`pdprgcopy`
	`pdprgrenew`	Updates to a HiRDB update version	`pdprgrenew`
HiRDB Datareplicator linkage	`pdrplstart`	Starts HiRDB Datareplicator linkage	`pdrplstart`
	`pdrplstop`	Stops HiRDB Datareplicator linkage	`pdrplstop`

Type	Command	Description	Reference
Directory Server linkage facility	`pdgrprfl`	Refreshes user information	`pdgrprfl`
	`pdusrchk`	Checks the directory server for user registration information	`pdusrchk`
Inner replica facility	`pddbchg`	Switches the replica status of replica RDAREAs	`pddbchg`
Updatable online reorganization	`pdorbegin`	Commits a database for online reorganization	`pdorbegin`
	`pdorcheck`	Checks application conditions for online reorganization	`pdorcheck`
	`pdorchg`	Changes the current RDAREA for online reorganization	`pdorchg`
	`pdorcreate`	Creates an environment for reflection processing for online reorganization	`pdorcreate`
	`pdorend`	Executes reflection processing for online reorganization	`pdorend`
Security audit	`pdaudbegin`	Starts acquisition of audit trails	`pdaudbegin`
	`pdaudend`	Stops acquisition of audit trails	`pdaudend`
	`pdaudrm`	Deletes audit trail files in shutdown status	`pdaudrm`
	`pdaudswap`	Swaps current audit trail files	`pdaudswap`
Connection security facility	`pdacunlck`	Releases the consecutive certification failure account lock state	`pdacunlck`
Transaction queuing facility	`pdtrnqing`	Starts and releases the transaction queuing facility	`pdtrnqing`
HiRDB External Data Access facility	`pddbadset`	Sets up HiRDB External Data Access Adapter	`pddbadset`
Real Time SAN Replication	`pdrisechk`	Checks the configuration of Real Time SAN Replication	`pdrisechk`
	`pdrisedbto`	Inherits a database in Real Time SAN Replication	`pdrisedbto`
	`pdriseset`	Sets the site status in Real Time SAN Replication	`pdriseset`
SQL trace acquisition	`pdclttrc`	Acquires SQL trace dynamically	`pdclttrc`
SQL object information display	`pdobils`	Displays statistical information about an SQL object buffer	`pdobils`

Type	Command	Description	Reference
SQL compilation	`pdcbl`	COBOL preprocessor	___ [1]
	`pdcpp`	C preprocessor	___ [1]
	`pdocb`	OOCOBOL preprocessor	___ [1]
	`pdocc`	C++ preprocessor	___ [1]
Database creation	`pdinit`	Database initialization utility	Chapter 3
	`pddef`	Database definition utility	Chapter 4
	`pdload`	Database load utility	Chapter 5
	`pdsql`	Interactive SQL execution utility[2]	Chapter 6
	`pddefrev`	Generates a definition SQL statement	`pddefrev`
Database operations	`pdmod`	Database structure modification utility	Chapter 7
	`pdrorg`	Database reorganization utility	Chapter 8
	`pdexp`	Dictionary import/export utility	Chapter 9
	`pdrbal`	Rebalancing utility	Chapter 10
	`pdreclaim`	Free page release utility	Chapter 11
	`pdpgbfon`	Global buffer residence utility	Chapter 12
	`pdconstck`	Integrity check utility	Chapter 13
Tuning	`pdstedit`	Statistics analysis utility	Chapter 14
	`pddbst`	Database condition analysis utility	Chapter 15
	`pdgetcst`	Optimizing information collection utility	Chapter 16
	`pdvwopt`	Access path display utility	Chapter 17
Database error handling	`pdcopy`	Database copy utility	`pdbkupls`
	`pdbkupls`	Displays information about backup files	Chapter 18
	`pdrstr`	Database recovery utility	Chapter 19
Plug-in-related	`pdplgrgst`	Registers and deletes plug-ins	`pdplgrgst`
	`pdplgset`	Sets up a plug-in	`pdplgset`
	`pdreginit`	Registry facility initialization utility	Chapter 20

—: Not applicable.

1 For details about `pdcbl`, `pdcpp`, `pdocb`, and `pdocc`, see the *HiRDB Version 8 UAP Development Guide*.

2 HiRDB SQL Executor is required in order to execute the interactive SQL execution utility.

B. Lock Mode During Command Execution

Some HiRDB commands place a resource in an appropriate lock mode automatically. Thus, when multiple programs (including UAPs) are executing concurrently, the user needs to check the command lock mode table and the UAP (SQL) lock table to determine the combination of lock modes. For details about the combinations of lock modes according to the SQL statements executed and the execution environment, see the *HiRDB Version 8 UAP Development Guide*.

The following lock modes are referenced in the tables in this appendix:

PR: Shared mode

EX: Lock mode

SR: Shared retrieval mode

SU: Shared update mode

PU: Protected update mode

↓: The lock mode during acquisition of check pending status is inherited.

—: Not locked

B.1 Lock mode for operation commands

Table B-1 shows the lock modes for the operation commands.

Table B-1: Lock modes for operation commands

Execution environment		Resource					
Command	**Option**	**Data dict table**	**Inner replica config**	**Replica group config**	**RDAREA status**	**RDAREA**	**Backup hold**
`pdbufls`	All	—	—	—	—	—	—
`pdclose`	None	—	—	—	EX	EX	—
`pddbchg`	None or `-w`	EX	EX	EX	EX	EX	—
`pddbfrz`	All	—	—	—	EX	EX	—
`pddbls`	`-a`	—	—	—	—	SR	—
	None, `-l`, `-b`, or `-o`	—	—	—	—	—	—

Execution environment		Resource					
Command	Option	Data dict table	Inner replica config	Replica group config	RDAREA status	RDAREA	Backup hold
`pdhold`	None or `-c`	—	—	—	EX	EX	—
	`-i`	—	—	—	EX	PR	—
	`-b`	—	—	—	EX	PR	—
	`-bw` or `-buw`	—	—	—	EX	PR	EX[2]
	`-bu`	—	—	—	EX	—	EX[2]
	`-s`	—	—	—	EX	EX	EX[2]
`pdopen`	None	—	—	—	EX	EX	—
`pdorbegin`	None	EX	EX	EX[3]	EX	EX	—
	`-u`	EX	—	—	EX	EX	—
`pdorchg`	None	EX	—	—	EX	EX	—
`pdorend`	None or `-z`	EX	EX	EX	EX	EX	—
	`-u`	EX	—	—	EX	EX	—
`pdrels`	None or `-o`	—	—	—	EX	PR/ —[1]	—

Legend:

config: configuration

dict: dictionary

[1] Lock is not placed when the resource is in updatable backup-hold status.

[2] Lock is not released until it is released by the `pdrels` command or the resource is placed in error shutdown status.

[3] Lock is not released until the `pdorchg` command terminates.

B.2 Lock mode for utilities

Tables B-2 through B-17 show the lock modes for each of the utilities.

The utilities listed as follows are not described here, because they do not provide the automatic lock function (the database definition utility is the same as for a definition SQL; see the *HiRDB Version 8 UAP Development Guide*):

- Database initialization utility (`pdinit`)
- Dictionary export/import utility (`pdexp`)
- Statistics analysis utility (`pdstedit`)
- Access path display utility (`pdvwopt`)

Table B-2: Lock modes for the database load utility

Execution environment	Resource						
	Table		Index		No-wait table	Resources management table	User LOB RDAREA
	RDAREA	Table ID	RDAREA	Index ID			
Data load by table	SU	EX	SU	EX	EX[2]	SR	SU
					PU[2]		
Data load by RDAREA	EX	—	SU	EX[1]	—	SR	EX
Data load of LOB data only	SR	PR	—	—	—	SR	EX

Note

When data loading is executed on a LOB column structure base table only, the lock mode is the same as for data loading by table when loading data by table or the same as for data loading by RDAREA when loading data by RDAREA.

[1] In the case of index loading by RDAREA, lock is not placed on a non-partitioning key index.

[2] During data deletion processing, the EX lock mode takes place. For other operations, if the `option` statement specifies `nowait=no`, the EX lock mode takes place; if it specifies `nowait=yes`, the PU lock mode takes place.

Table B-3: Lock modes for the database load utility (for shared tables)

Exe env	Resource								
	Table				Index			No-wait table	RDAREA mgmt block
	Updatbl back-end server		Ref-only back-end server		Shared RDAREA for updatbl back-end server	Shared RDAREA for ref-only back-end server	IX ID		
	Shared RDAREA	Tbl ID	Shared RDAREA	Tbl ID					
Data load	EX	EX	EX	EX	EX	EX	EX	EX	SR

Legend:

env: environment

Exe: Execution

IX: Index

load: loading

mgmt: management

Ref: Referencing

Tbl: Table

updatbl: updatable

Table B-4: Lock modes for the database structure modification utility

Exe env	Resource							
	DB structure mod utility	Data dict table	RDAREA status	RDAREA	Tbl	Res mgmt table	Inner replica config mgmt info	Replica group config mgmt info
RDAREA addition	EX	EX	EX^4	EX^4	—	—	—	—
RDAREA expansion	EX	EX	EX^4	PU(EX)^4	—	EX^4	—	—
RDAREA reinit	EX	EX	EX^4	EX^4	EX	—	—	—
RDAREA deletion	EX	EX	EX^4	EX^4	—	—	EX^4	EX^4

Exe env	**Resource**							
	DB structure mod utility	**Data dict table**	**RDAREA status**	**RDAREA**	**Tbl**	**Res mgmt table**	**Inner replica config mgmt info**	**Replica group config mgmt info**
Modification of data dictionary table attribute definition	EX	EX	—	EX[2]	—	—	—	—
Modification of RDAREA attribute	EX	EX	EX	EX	—	—	—	—
Changing an RDAREA from a HiRDB/Single Server structure to a HiRDB/Parallel Server structure	EX	EX	EX	EX	—	—	—	—
Changing to a back-end server structure[1]	EX	EX	EX	EX	—	—	—	—
Moving an RDAREA	EX	EX	EX[4]	EX[4]	—	—	—	—
Registration of a generation of a HiRDB file system area	EX	EX	—	—	—	—	—	—
Deletion of a generation of a HiRDB file system area	EX	EX	—	—	—	—	—	—
Definition of an RDAREA replica	EX	EX	EX[4]	EX[4]	—	—	EX[4]	EX[4]
Copying RDAREA configuration information	EX	EX	EX[4]	EX[4]	—	—	EX[4]	EX[4]
Integration of RDAREAs	EX	EX	EX[3,4]	EX[3,4]	—	—	EX[4]	EX[4]

Exe env	Resource							
	DB structure mod utility	Data dict table	RDAREA status	RDAREA	Tbl	Res mgmt table	Inner replica config mgmt info	Replica group config mgmt info
Auditor registration	EX	EX	—	—	—	—	—	—
Creation of an audit trail table	EX	EX	—	EX	—	EX	—	—

Legend:

config: configuration

DB: Database

Data dict table: Data dictionary table

Exe: Execution

info: information

mgmt: management

mod: modification

reinit: reinitialization

Res: Resources

Tbl: Table

1 This applies only to a HiRDB/Parallel Server; it is not applicable to a HiRDB/Single Server.

2 Applies only to the data dictionary RDAREA.

3 Applicable to both original and replica RDAREAs.

4 Applicable to both updatable back-end servers and referencing-only back-end servers in the case of shared RDAREAs.

Table B-5: Lock modes for the database reorganization utility

Execution environment		Resource							
		Table		Index		NW TBL	Res mgmt table	IX info file	User LOB RDAREA[1]
		RD AREA[1]	TBL ID	RD AREA[1]	IX ID				
Unloading by table (`-k unld`))	`unldenq=tblenq` is specified in `option` statement	SR	PR	SR[2]	—	—	SR	—	SR
	`unldenq=rdenq` is specified in `option` statement	PR	SR	PR[2]	—	—	SR	—	PR
	`unldenq=nowait` is specified in `option` statement	SR	—	SR[2]	—	SR	SR	—	SR
Unloading by RDAREA (`-k unld`)	`unldenq=tblenq` is specified in `option` statement	SR	PR	SR[2]	—	—	SR	—	SR
	`unldenq=rdenq` is specified in `option` statement	PR	SR	PR[2]	—	—	SR	—	PR
	`unldenq=nowait` is specified in `option` statement	SR	—	SR[2]	—	SR	SR	—	SR
Reloading by table (-k reld)		SU	EX	SU	EX	EX	SR	—	SU
Reloading by RDAREA (`-k reld`)		EX	—[3]	SU	EX[4]	—	SR	—	EX
Reorganization by table (`-k rorg`)		SU	EX	SU	EX	EX	SR	—	SU

Execution environment		Resource							
		Table		Index		NW TBL	Res mgmt table	IX info file	User LOB RDAREA[1]
		RD AREA[1]	TBL ID	RD AREA[1]	IX ID				
Reorganization by RDAREA (`-k rorg`)		EX	SR	SU	EX[4]	—	SR	—	EX
Batch creation of index (`-k ixmk`)		SR	SR	SU	EX	—	SR	—	—
Re-creation of index (`-k ixrc`)	`index` statement specified[5]	EX → SR[6]	— → SR[6]	SU	EX	—	SR	—	EX → —[6]
		SU → SR[6]	EX → SR[6]			EX → —[6]			SU → —[6]
	`idxname` statement specified	SU	EX	SU	EX	EX	SR	—	—
Reorganization of index (-k ixor)[7]		SR	PR	SU	EX	—	SR	—	—
				EX	—				
Batch creation of plug-in index (`-k ixmk`)		SR	SR	SU	EX	—	SR	EX	—
Re-creation of plug-in index (`-k ixrc`)	`index` statement specified[5]	EX → SR[6]	— → SR[6]	SU	EX	—	SR	EX	EX → —[6]
		SU → SR[6]	EX → SR[6]			EX → —[6]			SU → —[6]
	`idxname` statement specified	SU	EX	SU	EX	EX	SR	—	SU
Reloading of LOB data only (`-k reld`)		SR	PR	—	—	—	SR	—	EX
Reorganization of LOB data only (`-k rorg`)		SU	EX	SR[4]	—	EX	SR	—	SU

Legend:

RD AREA: RDAREA

TBL ID: Table ID

IX ID: Index ID

NW TBL: No-wait table

Res mgmt table: Resource management table

IX info file: Index information file

Note

When reload or reorganization is executed on a LOB column structure base table only, the lock mode is the same as for reloading or reorganization by table when loading or reorganizing data by table or the same as for reloading or reorganization by RDAREA when reloading or reorganizing data by RDAREA.

1

An RDAREA that is not the target of processing is not locked.

2

If data is unloaded in physical order, lock is not placed.

3

Lock is placed in the SR mode during unload processing.

4

Lock is not placed on a non-partitioned index during reload processing.

5

The upper row applies to a row-partitioned index; the lower row applies to a non-partitioned index.

6

The lock mode during unload processing on the index information (from `KFPL00725-I` to `KFPL00726-I`) is changed during index creation (from `KFPL00715-I` to `KFPL00716-I`).

7

The upper row applies when the index storage RDAREA is in command shutdown status; the lower row applies when it is not in command shutdown status.

Table B-6: Lock modes for the database reorganization utility (for shared tables)

Exe env	Resource									
	Table				Index			No-wait tbl	RD AREA mgmt block	IX info file
	Updatable backend server		Ref-only backend server		Shared RD AREA for updatbl backend server	Shared RD AREA for ref-only backend server	IX ID			
	Shared RD AREA	Tbl ID	Shared RD AREA	Tbl ID						
Unload by table (`-k unld`)	SR	PR	—	—	SR[1]	—	—	—	SR	—
Reload by table (`-k reld`)	EX	EX	EX	EX	EX	EX	EX	EX	SR	—
Reorg by table (`-k rorg`)	EX	EX	EX	EX	EX	EX	EX	EX	SR	—
Batch index creation (`-k ixmk`)	SR	SR	—	—	EX	EX	EX	—	SR	—
Re-crtn of index (`-k ixrc`, and `index` stmt specif)[2]	EX → SR	EX → SR	EX	EX	EX	EX	EX	EX → —	SR	—
Re-crtn of index (`-k ixrc`, and `idxname` stmt specif)	EX	EX	EX	EX	EX	EX	EX	EX	SR	—

Exe env	**Resource**									
	Table				**Index**			**No-wait tbl**	**RD AREA mgmt block**	**IX info file**
	Updatable backend server		**Ref-only backend server**		**Shared RD AREA for updatbl backend server**	**Shared RD AREA for ref-only backend server**	**IX ID**			
	Shared RD AREA	**Tbl ID**	**Shared RD AREA**	**Tbl ID**						
Index reorg (`-k ixor`)	SR	PR	—	—	EX	EX	EX	—	SR	—

Legend:

env: environment

Exe: Execution

info: information

IX: Index

mgmt: management

RD AREA: RDAREA

Re-crtn: Re-creation

Ref: referencing

Reorg: Reorganization

specif: specified

stmt: statement

Tbl: Table

updatbl: updatable

1 If data is unloaded in physical order, lock is not placed.

2 The lock mode for index information during an unload operation (from `KFPL00725-I` through `KFPL00726-I`) is changed when the index is loaded (from `KFPL00715-I` to `KFPL00716-I`).

Table B-7: Lock modes for the rebalancing utility

Execution environment	Resource					
	RDAREA	Table	No-wait table	Index	Rebalance*	Preprocessed table
Shared mode	SU	EX	EX	EX	EX	SR
Exclusive mode	SU	EX	EX	EX	—	SR

* For the rebalancing resources, lock is placed at the single server or front-end server.

Table B-8: Lock modes for free page release utility

Exe env	Resource									
	Table				No-wait table	Index				
	RD AREA	RD AREA mgmt block	Tbl ID	Free page rel util		RD AREA	RD AREA mgmt block	IX ID	Free page rel util	Wait for trxn compl
Releasing free pages from table	SU	SR	SU	EX	SR	—	—	—	—	—
Releasing free segments from table	EX	SR	—	EX	—	—	—	—	—	—
Releasing free pages from index	—	—	—	—	SR	SU	SR	EX	EX	PR
Releasing free segments from index	—	—	SR	—	—	EX	SR	—	EX	—

Legend:

RD AREA: RDAREA

RD AREA mgmt block: RDAREA management block

Tbl ID: Table ID

Free page rel util: Free page release utility

IX ID: Index ID

IX info file: Index information file

Wait for trxn compl: Wait for transaction completion

Table B-9: Lock modes for the free page release utility (for shared tables)

Exe env	Resource											
	Table						Index					
	Updatbl backend server		Ref-only backend server		RMB	Free page rel util	SR for UBS	SR for RBS	RMB	IX ID	Free page rel util	Wait for trxn compl
	SR	Tbl ID	SR	Tbl ID								
Release free pages from table	EX	EX	EX	EX	SR	EX	—	—	—	—	—	—
Release free segmts from table	EX	EX	EX	EX	SR	EX	—	—	—	—	—	—
Release free pages from index	—	SR	—	—	—	—	EX	EX	SR	EX	EX	PR
Release free segmts from index	—	SR	—	—	—	—	EX	EX	SR	—	EX	—

Legend:

compl: completion

env: environment

Exe: Execution

IX: Index

mgmt: management

rel: release

segmts: segments

RMB: RDAREA management block

SR: Shared RDAREA

SR for UBS: Shared RDAREA for updatable back-end server

SR for RBS: Shared RDAREA for referencing-only back-end server

Tbl: Table

Updatbl: Updatable

util: utility

Table B-10: Lock modes for the global buffer residence utility

Exe env	**Resource**					
	Table			**Index**		
	RDAREA	**RDAREA management block**	**No-wait table**	**RDAREA**	**RDAREA management block**	**Index ID**
Reading data page (`-k table`)	SR	SR	SR	—	—	—
Reading index page (`-k index`)	—	—	SR	SR	SR	SR

Legend:

env: environment

Exe: Execution

Table B-11: Lock modes for the global buffer residence utility (for shared tables)

Exe env	**Resource**											
	Table						**Index**					
	Updatable backend server			**Referencing-only backend server**			**Updatable backend server**			**Referencing-only backend server**		
	R	**RMB**	**No-wait tbl**	**R**	**RMB**	**No-wait tbl**	**R**	**RMB**	**IX ID**	**R**	**RMB**	**IX ID**
Reading data page (`-k table`)	SR	SR	SR	SR	SR	SR	—	—	—	—	—	—
Reading index page (`-k index`)	—	—	SR	—	—	SR	SR	SR	SR	SR	SR	SR

Legend:

env: environment

Exe: Execution

IX: Index

R: RDAREA

RMB: RDAREA management block

Table B-12: Lock modes for the database condition analysis utility

Execution environment	**Resource**					
	RDAREA			**Resources management table**	**Table**	**No-wait table**[1]
	For table	**For index**	**Specified RDAREA**			
Analysis by RDAREA	—	—	SR	SR	—	SR
Analysis by table	SR	—	—	SR	SR	SR
Analysis by index	—	SR	—	SR	SR	SR

Execution environment	Resource					
	RDAREA			Resources management table	Table	No-wait table[1]
	For table	For index	Specified RDAREA			
Condition analysis result accumulation facility	SR	SR	SR	SR	SR	—
Facility for predicting reorganization time[2]	SR	SR	SR	SR	SR	SR

[1] A no-wait table is locked only when the `-d` option is specified.

[2] This lock is applicable when the `-m` option is specified.

Table B-13: Lock modes for the optimizing information collection utility

Execution environment	Resource							
	RDAREA		Resources management table		Table	No-wait table	Data dictionary table	
	For table	For index	For table	For index			Row	Key
Acquisition of optimization information	SR	SR	SR	SR	SR/ SU[1]	SR	PR[2]/ EX[1]	PR[2]/EX[1]
Deletion of optimization information	—	—	—	—	SU[1]	—	PR[2]/ EX[1]	PR[2]/EX[1]

[1] Lock is placed temporarily during data dictionary table retrieval.

[2] Lock is placed during data dictionary table update processing.

Table B-14: Lock modes for the database copy utility

Execution environment		**Resource**			
		Data dictionary table	**RDAREA**	**Directory block 6**	**Resources management table**
`-M` option	`x`	PR	PR	—	—
	`r`	PR	PR	—	—
	`s`	—	—	PR	SR

Table B-15: Lock modes for the database recovery utility

Execution environment	**Resource**	
	Data dictionary table	**RDAREA**
Recovery of RDAREA	EX*	EX

* Lock is placed if the master directory RDAREA is to be recovered.

Table B-16: Lock modes for the registry facility initialization utility

Execution Environment	**Resource**			
	Database structure modification utility	**Data dictionary table**	**RDAREA condition**	**RDAREA**
Addition of registry RDAREAs, registry LOB RDAREAs, and registry management tables (`-k all`)	EX	EX	EX	EX

Table B-17: Lock modes for the integrity check utility

Execution environment		Resource								
		User table				Data dictionary table				
		RDAREA		TBL	NW TBL	Row	PP TBL	TBL	RDAREA	
		For TBL	For IX						For TBL	For IX
Integrity checking (`-k check`)	When searching data dictionary table	—	—	—	—	PR	PR	SR	SR	SR
	When acquiring RDAREA check pending status*	SR	SR	PR	—	—	—	—	—	—
	When performing integrity checking	↓	↓	↓	—	—	—	—	—	—
	When setting check pending status	SU	SU	EX	EX	EX	PR	SU	SU	SU
Forced setting of check pending status (`-k set`) or forced release of check pending status (`-k release`)	When searching data dictionary table	—	—	—	—	PR	PR	SR	SR	SR
	When acquiring RDAREA check pending status*	SR	SR	PR	—	—	—	—	—	—
	When setting or releasing check pending status	SU	↓	EX	EX	EX	PR	SU	SU	SU

Legend:

For TBL: For table

For IX: For index

NW TBL: No-wait table

PP TBL: Preprocessing table

TBL: Table

* Lock is placed by `UNTIL DISCONNECT` (lock is maintained until `DISCONNECT` is executed after the setting of check pending status is completed).

C. RDAREA Status During Command Execution

C.1 RDAREA status transitions

Tables C-1 and C-2 describe the RDAREA status transitions during execution of operation commands.

Table C-1: RDAREA status transitions (1/2)

RDAREA status (before transition)	**RDAREA status (after transition)**					
	Open	**Shutdown**	**Closed**	**Shutdown and closed**	**Reference -possible shutdown**	**Reference-possible shutdown and closed**
Open[1]	—	`pdhold`	Transaction termination[3]	`pdhold -c`	`pdhold -i`	N
Shutdown	`pdrels` or `pdrels -o`	—	N	`pdclose`	`pdhold -i`	N
Closed	`pdopen`, `pdrels -o`, or transaction startup[2]	N	—	`pdhold` or `pdhold -c`	N	`pdhold -i`
Shutdown and closed	`pdrels -o`	`pdopen`	`pdrels`	—	N	`pdhold -i`
Ref-possible shutdown	`pdrels` or `pdrels -o`	N	N	N	—	`pdclose` or transaction termination[3]
Ref-possible shutdown and closed	N	N	`pdrels`	`pdhold` or `pdhold -c`	`pdopen` or transaction startup[2]	—
Ref-possible backup-hold	`pdrels` or `pdrels -o`	N	N	N	N	N
Ref-possible backup-hold and closed	`pdrels -o`	N	`pdrels`	N	N	N
Updatable backup-hold	`pdrels` or `pdrels -o`	N	N	N	N	N

RDAREA status (before transition)	RDAREA status (after transition)					
	Open	Shutdown	Closed	Shutdown and closed	Reference -possible shutdown	Reference-possible shutdown and closed
Updatable backup-hold and closed	pdrels -o	N	`pdrels`	N	N	N
Synchro shutdown	`pdrels` or `pdrels -o`	N	N	N	N	N
Synchro shutdown and closed	`pdrels -o`	N	`pdrels`	N	N	N
Online reorganization hold	`pdorbegin -u`[4], `pdorend`[4], `pdorend -z`[4], or `pdorend -u`	`pdhold`	N	`pdhold -c`, `pdorbegin -u`[5], `pdorend`[5], or `pdorend -z`[5]	N	N
Online reorganization hold and closed	N	N	`pdorbegin -u`[4], `pdorend`[4], `pdorend -z`[4], or `pdorend -u`	`pdhold`, `pdorbegin -u`[5], `pdorend`[5], or `pdorend -z`[5]	N	N

Legend:

— : Not applicable

N: Cannot be in this RDAREA status after the transition

Ref: Reference

Synchro: Synchronization

[1] The `pdclose` command will result in an error because the RDAREA is not shut down.

[2] This status changes if the RDAREA open timing is either `DEFER` or `SCHEDULE`.

[3] This status changes if the RDAREA open timing is `SCHEDULE`.

[4] This status change is applicable to original RDAREAs.

5 This status change is applicable to replica RDAREAs.

Table C-2: RDAREA status transitions (2/2)

RDAREA status (before change)	RDAREA status (after change)							
	RPH	RPHC	UH	UHC	SS	SSC	ORH	ORHC
Open[1]	`pdhold -b`	N	`pdhold -b -u`	N	`pdhold -s`	N	`pdorbegin`[6]	N
Shutdown	N	N	N	N	N	N	`pdrels`	N
Closed	`pdhold -b`[5]	N	`pdhold -b -u`[5]	N	N	N	`pdorbegin`[5, 6]	`pdorbegin`[3, 6]
Shutdown and closed	N	`pdhold -b`[2]	N	`pdhold -b -u`[2]	N	N	`pdrels -o`	`pdrels` or `pdorbegin`[7]
Ref-possible shutdown	N	N	N	N	N	N	N	N
Ref-possible shutdown and closed	N	N	N	N	N	N	N	N
Ref-possible backup-hold	—	trxn term[3] or `pdclose`[4]	N	N	`pdhold -s`	N	N	N
Ref-possible backup-hold and closed	Trxn startup[2] or `pdopen`[4]	—	N	N	N	N	N	N
Updatable backup-hold	N	N	—	Trxn term[3]	N	N	N	N
Updatable backup-hold and closed	N	N	Trxn startup[2]	—	N	N	N	N
Synchro shutdown	N	N	N	N	—	`pdclose` or trxn term[3]	N	N

RDAREA status (before change)	RDAREA status (after change)							
	RPH	**RPHC**	**UH**	**UHC**	**SS**	**SSC**	**ORH**	**ORHC**
Synchro shutdown and closed	N	N	N	N	`pdopen` or trxn startup[2]	—	N	N
Online reorg hold	N	N	N	N	N	N	—	`pdclose` or trxn term
Online reorg hold and closed	N	N	N	N	N	N	`pdopen`, trxn startup, or `pdorchg`[7]	—

Legend:

N: Cannot be in this RDAREA status after the transition

RPH: Reference-possible backup hold

RPHC: Reference-possible backup hold and closed

UH: Updatable backup-hold

UHC: Updatable backup-hold and closed

SS: Synchronization shutdown

SSC: Synchronization shutdown and closed

ORH: Online reorganization hold

ORHC: Online reorganization hold and closed

—: Not applicable

Ref: Reference

reorg: reorganization

Synchro: Synchronization

term: termination

Trxn: Transaction

[1] The `pdclose` command will result in an error because the RDAREA is not shut down.

2 This status changes if the RDAREA open timing is either `DEFER` or `SCHEDULE`.

3 This status changes if the RDAREA open timing is `SCHEDULE`.

4 This status changes if the RDAREA is not in reference-possible backup hold (update WAIT mode).

5 This status changes if the RDAREA open timing is `DEFER`.

6 This status change is applicable to original RDAREAs.

7 This status change is applicable to replica RDAREAs.

C.2 Availability of utility or UAP execution depending on RDAREA status

Whether or not a utility or UAP can be executed depends on the RDAREA's open timing and status. Tables C-3 through C-8 describe the availability of utility or UAP execution depending on the RDAREA status.

Table C-3: Availability of utility or UAP execution depending on RDAREA status (when open timing is set to INITIAL) (1/3)

Utility or UAP			No shutdown		Command shutdown		Ref-possible shutdown	
			Open	Closed	Open	Closed	Open	Closed
Database structure modification utility	RDAREA addition		—	—	—	—	—	—
	RDAREA deletion		N	N	N	Y	N	Y
	RDAREA re-initialization		N	N	N	Y	N	Y
	RDAREA expansion^3		Y	N	Y	N	Y	N
	Modification of RDAREA attribute		N	N	N	Y	N	Y
	Definition of an RDAREA replica	Original RDAREA	Y	Y	Y	Y	Y	Y
		Replica RDAREA	—	—	—	—	—	—
	Copying of RDAREA configuration information	Copy source	Y	Y	Y	Y	Y	Y
		Copy target	N	N	N	Y	N	Y
	Integration of RDAREAs	Original RDAREA	N	N	N	Y	N	Y
		Replica RDAREA	N	N	N	Y	N	Y
Database load utility^3			Y^4	N	Y^4	N	Y^4	N
Database reorganization utility^3	Reorganization (`-k rorg`)		Y$^{2, 4}$	N	Y	N	Y$^{2, 4}$	N
	Unloading (`-k unld`)		Y	N	Y	N	Y	N
	Reloading (`-k reld`)		Y$^{2, 4}$	N	Y	N	Y$^{2, 4}$	N
	Batch creation of index (`-k ixmk`)		Y	N	Y	N	Y	N
	Re-creation of index (`-k ixrc`)		Y	N	Y	N	Y	N
	Re-organization of index (`-k ixor`)		Y	N	Y	N	Y	N

Utility or UAP		No shutdown		Command shutdown		Ref-possible shutdown	
		Open	Closed	Open	Closed	Open	Closed
Free page release utility[3]		Y	N	Y	N	Y	N
Global buffer residence utility		Y	N	N	N	Y	N
Rebalancing utility		Y	N	Y	N	Y	N
Database copy utility	`-M x`	N	Y	N	Y	N	Y
	`-M r`	Y	Y	Y	Y	Y	Y
	`-M s`	Y	Y	Y	Y	Y	Y
Database recovery utility		N	N	N	Y	N	N
Database condition analysis utility		Y	N	Y	N	Y	N
Optimizing information collection utility		Y	N	N	N	Y	N
Dictionary import/export utility		Y	N	N	N	N	N
Database definition utility		Y	N	N	N	N	N
Integrity check utility		Y	N	N	N	N	N
Reflection processing for online reorganization (`pdorend` command)		N	N	N	N	N	N
UAP		Y	N	N	N	Y[1]	N

Legend:

Y: Can be executed

N: Cannot be executed

—: Not applicable

config: configuration

Ref: reference

[1] Only a UAP that executes referencing operations can be executed.

[2] A utility or UAP cannot be executed on a falsification prevented table.

[3] This cannot be executed if the original RDAREA in online reorganization hold status in the same replica group is the current RDAREA.

[4]For whether or not the referencing table can be placed in check pending status by executing the utility on the referenced table, see *C.3 Whether or not the check pending status can be set*.

Table C-4: Availability of utility or UAP execution depending on RDAREA status (when open timing is set to INITIAL) (2/3)

Utility or UAP			Reference-possible backup hold		Updatable backup-hold	Error shutdown	
			Open	Closed	Open	Open	Closed
Database structure modification utility	RDAREA addition		—	—	—	—	—
	RDAREA deletion		N	N	N	N	Y
	RDAREA re-initialization		N	N	N	N	Y
	RDAREA expansion[4]		N	N	N	N	N
	RDAREA modification		N	N	N	N	N
	Definition of an RDAREA replica	Original RDAREA	Y	Y	Y	Y	Y
		Replica RDAREA	—	—	—	—	—
	Copying of RDAREA configuration information	Copy source	Y	Y	Y	Y	Y
		Copy target	N	N	N	N	N
	Integration of RDAREAs	Original RDAREA	N	N	N	N	N
		Replica RDAREA	N	N	N	N	N
Database load utility[4]			N	N	N	N	N

Utility or UAP		Reference-possible backup hold		Updatable backup-hold	Error shutdown	
		Open	Closed	Open	Open	Closed
Database reorganization utility[4]	Reorganization (`-k rorg`)	N	N	N	N	N
	Unloading (`-k unld`)	Y	N	Y	N	N
	Reloading (`-k reld`)	N	N	N	N	N
	Batch creation of index (`-k ixmk`)	N	N	N	N	N
	Re-creation of index (`-k ixrc`)	N	N	N	N	N
	Re-organization of index (`-k ixor`)	N	N	N	N	N
Free page release utility[4]		N	N	N	N	N
Global buffer residence utility		Y	N	Y	N	N
Rebalancing utility		N	N	N	N	N
Database copy utility	`-M x`	N	Y	N	N	Y
	`-M r`	Y	Y	Y[1]	N	N
	`-M s`	Y	Y	Y	Y	Y
Database recovery utility		N	N	N	N	Y
Database condition analysis utility		Y	N	Y	N	N
Optimizing information collection utility		Y[2]	N	Y[2]	N	N
Dictionary import/export utility		N	N	Y	N	N
Database definition utility		N	N	Y	N	N
Integrity check utility		Y	N	Y	N	N
Reflection processing for online reorganization (`pdorend` command)		N	N	N	N	N
UAP		Y[3]	N	Y	N	N

Legend:

Y: Can be executed

N: Cannot be executed

—: Not applicable

[1] A utility or UAP cannot be executed if an updating transaction is being executed.

[2] A utility or UAP cannot be executed if the dictionary RDAREA is in reference-possible backup hold status.

[3] Only a UAP that executes referencing operations can be executed.

[4] This cannot be executed if the original RDAREA in online reorganization hold status in the same replica group is the current RDAREA.

Table C-5: Availability of utility or UAP execution depending on RDAREA status (when open timing is set to INITIAL) (3/3)

Utility or UAP			**No-log shutdown**		**Synchronizatio n shutdown**		**Online reorganization hold**	
			Open	**Closed**	**Open**	**Closed**	**Open**	**Closed**
Database structure modification utility	RDAREA addition		—	—	—	—	—	—
	RDAREA deletion		N	Y	N	N	N	N
	RDAREA re-initialization		N	Y	N	Y	N	Y
	RDAREA expansion[8]		N	N	N	N	Y	N
	RDAREA modification		N	N	N	N	N	N
	Definition of an RDAREA replica	Original RDAREA	Y	Y	Y	Y	Y	Y
		Replica RDAREA	—	—	—	—	—	—
	Copying of RDAREA configuratin information	Copy source	Y	Y	Y	Y	Y	Y
		Copy target	N	N	N	N	N	N
	Integration of RDAREAs	Original RDAREA	N	N	N	N	N	N
		Replica RDAREA	N	N	N	N	N	N
Database load utility[8]			N	N	N	N	Y[3, 10]	N

Utility or UAP		No-log shutdown		Synchronization shutdown		Online reorganization hold	
		Open	Closed	Open	Closed	Open	Closed
Database reorganization utility[8]	Reorganization (`-k rorg`)	N	N	N	N	Y[4, 5, 10]	N
	Unloading (`-k unld`)	N	N	N	N	Y[5]	N
	Reloading (`-k reld`)	N	N	N	N	Y[4, 5, 10]	N
	Batch creation of index (`-k ixmk`)	N	N	N	N	Y[5]	N
	Re-creation of index (`-k ixrc`)	N	N	N	N	Y[5]	N
	Re-organization of index (`-k ixor`)	N	N	N	N	Y[5]	N
Free page release utility[8]		N	N	N	N	Y	N
Global buffer residence utility		N	N	N	N	Y	N
Rebalancing utility		N	N	N	N	N	N
Database copy utility	`-M x`	N	Y	N	N	N	Y
	`-M r`	N	N	N	N	Y	Y
	`-M s`	Y	Y	N	N	Y	Y
Database recovery utility		N	Y	N	Y	N	N
Database condition analysis utility		N	N	N	N	Y[9]	N
Optimizing information collection utility		N	N	N	N	N[7]	N
Dictionary import/export utility		N	N	—	—	—	—
Database definition utility		N	N	—	—	—	—
Integrity check utility		N	N	Y[11]	N[12]	Y	N
Reflection processing for online reorganization (`pdorend` command)		N	N	N	N	Y	N
UAP		Y[1]	N	Y[2]	Y[2]	Y[6]	Y[6]

Legend:

Y: Can be executed

N: Cannot be executed

—: Not applicable

config: configuration

1 Only the `PURGE TABLE` statement can be executed.

2 A utility or UAP will be placed in lock-release wait status.

3 The utility cannot be executed if the creation mode, no-log mode (or pre-update log acquisition mode), or the number of local buffer sectors for batch output is specified for a replica RDAREA, or on a table for which a LOB column has been defined.

4 A utility or UAP cannot be executed on a falsification prevented table.

5 This cannot be executed on a replica RDAREA.

6 Only an SQL statement in the log acquisition mode that accesses the current RDAREA can be executed (except in the case of the `CREATE TABLE`, `CREATE INDEX`, `ALTER TABLE`, `DROP TABLE`, `DROP INDEX`, `DROP SCHEMA`, `PURGE TABLE`, or `LOCK` statement).

7 The utility can be executed if `-c lvl1` is specified for the replica RDAREA.

8 This cannot be executed if the original RDAREA in online reorganization hold status in the same replica group is the current RDAREA.

9 This cannot be executed when the `-s` option is specified.

10 For whether or not the referencing table can be placed in check pending status by executing the utility on the referenced table, see *C.3 Whether or not the check pending status can be set*.

11 A utility or UAP is placed in lock-release wait status until the shutdown status is released. After the shutdown status is released, the utility or UAP can be executed.

12 A utility or UAP is placed in lock-release wait status until the shutdown status is released. After the shutdown status is released, the utility or UAP can be executed by opening the RDAREA, which has been closed.

Table C-6: Availability of utility or UAP execution depending on RDAREA status (when open timing is set to DEFER or SCHEDULE) (1/3)

Utility or UAP			No shutdown		Command shutdown		Ref-possible shutdown	
			Open	Closed	Open	Closed	Open	Closed
Database structure modification utility	RDAREA addition		—	—	—	—	—	—
	RDAREA deletion		N	N	N	Y	N	Y
	RDAREA re-initialization		N	N	N	Y	N	Y
	RDAREA expansion[4]		Y	Y	Y	N	Y	N
	Modification of RDAREA attribute		N	N	N	Y	N	Y
	Definition of an RDAREA replica	Original RDAREA	Y	Y	Y	Y	Y	Y
		Replica RDAREA	—	—	—	—	—	—
	Copying of RDAREA configuration information	Copy source	Y	Y	Y	Y	Y	Y
		Copy target	N	N	N	Y	N	Y
	Integration of RDAREAs	Original RDAREA	N	N	N	Y	N	Y
		Replica RDAREA	N	N	N	Y	N	Y
Database load utility[4]			Y[5]	Y[5]	Y[5]	N	Y[5]	Y[5]
Database reorganization utility[4]	Reorganization (`-k rorg`)		Y[3, 5]	Y[3, 5]	Y	N	Y[3, 5]	Y[3, 5]
	Unloading (`-k unld`)		Y	Y	Y	N	Y	Y
	Reloading (`-k reld`)		Y[3, 5]	Y[3, 5]	Y	N	Y[3, 5]	[3, 5]
	Batch creation of index (`-k ixmk`)		Y	Y	Y	N	Y	Y
	Re-creation of index (`-k ixrc`)		Y	Y	Y	N	Y	Y
	Re-organization of index (`-k ixor`)		Y	Y	Y	N	Y	Y

Utility or UAP		No shutdown		Command shutdown		Ref-possible shutdown	
		Open	Closed	Open	Closed	Open	Closed
Free page release utility[4]		Y	Y	Y	N	Y	Y
Global buffer residence utility		Y	Y	N	N	Y	Y
Rebalancing utility		Y	Y	Y	N	Y	Y
Database copy utility	`-M x`	N	Y	N	Y	N	Y
	`-M r`	Y	Y	Y	Y	Y	Y
	`-M s`	Y	Y	Y	Y	Y	Y
Database recovery utility		N	N	N	Y	N	N
Database condition analysis utility		Y	Y	Y	N	Y	Y
Optimizing information collection utility		Y	Y	N	N	Y	Y
Dictionary import/export utility[1]		—	—	—	—	—	—
Database definition utility		Y	N	N	N	N	N
Integrity check utility		Y	Y	N	N	N	N
Reflection processing for online reorganization (`pdorend` command)		N	N	N	N	N	N
UAP		Y	Y	N	N	Y[2]	Y[2]

Legend:

Y: Can be executed

N: Cannot be executed

— : Not applicable

config: configuration

Ref: Reference

[1] This utility is not applicable because the RDAREA's open timing can never be `DEFER` or `SCHEDULE`.

[2] Only a UAP that executes referencing operations can be executed.

[3] A utility or UAP cannot be executed on a falsification prevented table.

[4] This cannot be executed if the original RDAREA in online reorganization hold status in the same replica group is the current RDAREA.

[5]For whether or not the referencing table can be placed in check pending status by executing the utility on the referenced table, see *C.3 Whether or not the check pending status can be set*.

Table C-7: Availability of utility or UAP execution depending on RDAREA status (when open timing is set to DEFER or SCHEDULE) (2/3)

Utility or UAP			Ref-possible backup hold		Updatable backup-hold		Error shutdown	
			Open	Closed	Open	Closed	Open	Closed
Database structure modification utility	RDAREA addition		—	—	—	—	—	—
	RDAREA deletion		N	N	N	N	N	Y
	RDAREA re-initialization		N	N	N	N	N	Y
	RDAREA expansion[5]		N	N	N	N	N	N
	RDAREA modification		N	N	N	N	N	N
	Definition of an RDAREA replica	Original RDAREA	Y	Y	Y	Y	Y	Y
		Replica RDAREA	—	—	—	—	—	—
	Copying of RDAREA configuration information	Copy source	Y	Y	Y	Y	Y	Y
		Copy target	N	N	N	N	N	N
	Integration of RDAREAs	Original RDAREA	N	N	N	N	N	N
		Replica RDAREA	N	N	N	N	N	N
Database load utility[5]			N	N	N	N	N	N

Utility or UAP		Ref-possible backup hold		Updatable backup-hold		Error shutdown	
		Open	Closed	Open	Closed	Open	Closed
Database reorganization utility[5]	Reorganization (`-k rorg`)	N	N	N	N	N	N
	Unloading (`-k unld`)	Y	Y	Y	Y	N	N
	Reloading (`-k reld`)	N	N	N	N	N	N
	Batch creation of index (`-k ixmk`)	N	N	N	N	N	N
	Re-creation of index (`-k ixrc`)	N	N	N	N	N	N
	Re-organization of index (`-k ixor`)	N	N	N	N	N	N
Free page release utility[5]		N	N	N	N	N	N
Global buffer residence utility		Y	Y	Y	Y	N	N
Rebalancing utility		N	N	N	N	N	N
Database copy utility	`-M x`	N	Y	N	Y	N	Y
	`-M r`	Y	Y	Y[1]	Y[1]	N	N
	`-M s`	Y	Y	Y	Y	Y	Y
Database recovery utility		N	N	N	N	N	Y
Database condition analysis utility		Y	Y	Y	Y	N	N
Optimizing information collection utility		Y[3]	Y[3]	Y[3]	Y[3]	N	N
Dictionary import/export utility[2]		—	—	—	—	—	—
Database definition utility		N	N	Y	Y	N	N
Integrity check utility		N	N	Y	Y	N	N
Reflection processing for online reorganization (`pdorend` command)		N	N	N	N	N	N
UAP		Y[4]	Y[4]	Y	Y	N	N

Legend:

Y: Can be executed

N: Cannot be executed

—: Not applicable

config: configuration

Ref: Reference

[1] A utility or UAP cannot be executed if an updating transaction is being executed.

[2] This utility is not applicable because the RDAREA's open timing can never be `DEFER` or `SCHEDULE`.

[3] A utility or UAP cannot be executed if the dictionary RDAREA is in reference-possible backup hold status.

[4] Only a UAP that executes referencing operations can be executed.

[5] This cannot be executed if the original RDAREA in online reorganization hold status in the same replica group is the current RDAREA.

Table C-8: Availability of utility or UAP execution depending on RDAREA status (when open timing is set to DEFER or SCHEDULE) (3/3)

Utility or UAP			No-log shutdown		Synchronization shutdown		Online reorganization hold	
			Open	Closed	Open	Closed	Open	Closed
Database structure modification utility	RDAREA addition		—	—	N	N	—	—
	RDAREA deletion		N	Y	N	N	N	N
	RDAREA re-initialization		N	Y	N	Y	N	Y
	RDAREA expansion^{9}		N	N	N	N	Y	N
	RDAREA modification		N	N	N	N	N	N
	Definition of an RDAREA replica	Original RDAREA	Y	Y	Y	Y	Y	Y
		Replica RDAREA	—	—	—	—	—	—
	Copying of RDAREA configuration information	Copy source	Y	Y	Y	Y	Y	Y
		Copy target	N	N	N	N	N	N
	Integration of RDAREAs	Original RDAREA	N	N	N	N	N	N
		Replica RDAREA	N	N	N	N	N	N
Database load utility^{9}			N	N	N	N	Y$^{5, 11}$	Y$^{5, 11}$

Utility or UAP		No-log shutdown		Synchronization shutdown		Online reorganization hold	
		Open	Closed	Open	Closed	Open	Closed
Database reorganization utility[9]	Reorganization (`-k rorg`)	N	N	N	N	Y[4, 6, 11]	Y[4, 6, 11]
	Unloading (`-k unld`)	N	N	N	N	Y[6]	Y[6]
	Reloading (`-k reld`)	N	N	N	N	Y[4, 6, 11]	Y[4, 6, 11]
	Batch creation of index (`-k ixmk`)	N	N	N	N	Y[6]	Y[6]
	Re-creation of index (`-k ixrc`)	N	N	N	N	Y[6]	Y[6]
	Re-organization of index (`-k ixor`)	N	N	N	N	Y[6]	Y[6]
Free page release utility[9]		N	N	N	N	Y	Y
Global buffer residence utility		N	N	N	N	Y	Y
Rebalancing utility		N	N	N	N	N	N
Database copy utility	`-M x`	N	Y	N	N	N	Y
	`-M r`	N	N	N	N	Y	Y
	`-M s`	Y	Y	N	N	N	Y
Database recovery utility		N	Y	N	Y	N	N
Database condition analysis utility		N	N	N	N	Y[10]	Y[10]
Optimizing information collection utility		N	N	N	N	N[8]	N
Dictionary import/export utility[1]		—	—	—	—	—	—
Database definition utility		N	N	—	—	—	—
Integrity check utility		N	N	Y[12]	Y[12]	Y	Y
Reflection processing for online reorganization (`pdorend` command)		N	N	N	N	Y	Y
UAP		Y[2]	Y[2]	Y[3]	Y[3]	Y[7]	Y[7]

Legend:

Y: Can be executed

N: Cannot be executed

—: Not applicable

config: configuration

[1] This utility is not applicable because the RDAREA's open timing can never be `DEFER` or `SCHEDULE`.

[2] Only the `PURGE TABLE` statement can be executed.

[3] A utility or UAP will be placed in lock-release wait status.

[4] A utility or UAP cannot be executed on a falsification prevented table.

[5] The utility cannot be executed if the creation mode, no-log mode (or pre-update log acquisition mode), or the number of local buffer sectors for batch output is specified for a replica RDAREA, or on a table for which a LOB column has been defined.

[6] This cannot be executed on a replica RDAREA.

[7] Only an SQL statement in the log acquisition mode that accesses the current RDAREA can be executed (except in the case of the `CREATE TABLE`, `CREATE INDEX`, `ALTER TABLE`, `DROP TABLE`, `DROP INDEX`, `DROP SCHEMA`, `PURGE TABLE`, or `LOCK` statement).

[8] The utility can be executed if `-c lvl1` is specified for the replica RDAREA.

[9] This cannot be executed if the original RDAREA in online reorganization hold status in the same replica group is the current RDAREA.

[10] This cannot be executed when the `-s` option is specified.

[11] For whether or not the referencing table can be placed in check pending status by executing the utility on the referenced table, see Appendix *C.3 Whether or not the check pending status can be set*.

[12] A utility or UAP is placed in lock-release wait status until the shutdown status is released. After the shutdown status is released, the utility or UAP can be executed.

C.3 Whether or not the check pending status can be set

Whether or not a utility can be executed on a referenced table in order to place its referencing table in check pending status depends on the RDAREA open timing and the RDAREA's status. Tables C-9 through C-14 show whether or not the check pending status can be set by a utility depending on the RDAREA's status.

Table C-9: Whether or not the check pending status can be set by a utility depending on the RDAREA's status (when open timing is set to INITIAL) (1/3)

Utility	**No shutdown**		**Command shutdown**		**Ref-possible shutdown**	
	Open	**Closed**	**Open**	**Closed**	**Open**	**Closed**
Database structure modification utility	Y	S	S	S	S	S
Database load utility	Y	N	Y	N	Y	N
Database reorganization utility	Y	N	Y	N	Y	N
Reflection processing for online reorganization (`pdorend` command)	Y	S	S	S	S	S

Legend:

Y: Can be set to check pending status.

S: Cannot be set to check pending status. The utility skips this processing and executes the next processing.

N: Cannot be set to check pending status.

Table C-10: Whether or not the check pending status can be set by a utility depending on the RDAREA's status (when open timing is set to INITIAL) (2/3)

Utility	**Reference-possible backup hold**		**Updatable backup-hold**	**Error shutdown**	
	Open	**Closed**	**Open**	**Open**	**Closed**
Database structure modification utility	S	S	Y	S	S
Database load utility	N	N	Y	N	N
Database reorganization utility	N	N	Y	N	N
Reflection processing for online reorganization (`pdorend` command)	S	S	Y	S	S

Legend:

Y: Can be set to check pending status.

S: Cannot be set to check pending status. The utility skips this processing and executes the next processing.

N: Cannot be set to check pending status.

Table C-11: Whether or not the check pending status can be set by a utility depending on the RDAREA's status (when open timing is set to INITIAL) (3/3)

Utility	**No-log shutdown**		**Synchronization shutdown**		**Online reorganization hold**	
	Open	**Closed**	**Open**	**Closed**	**Open**	**Closed**
Database structure modification utility	S	S	S	S	Y	S
Database load utility	N	N	W[1]	N[2]	Y	N
Database reorganization utility	N	N	W[1]	N[2]	Y	N
Reflection processing for online reorganization (`pdorend` command)	S	S	S	S	Y	S

Legend:

Y: Can be set to check pending status.

W: Waits until the shutdown status is released and then sets the check pending status.

S: Cannot be set to check pending status. The utility skips this processing and executes the next processing.

N: Cannot be set to check pending status.

[1] A utility or UAP is placed in lock-release wait status until the shutdown status is released. After the shutdown status is released, the utility or UAP can be executed.

[2] A utility or UAP is placed in lock-release wait status until the shutdown status is released. After the shutdown status is released, the utility or UAP can be executed by opening the RDAREA, which has been closed.

Table C-12: Whether or not the check pending status can be set by a utility depending on the RDAREA's status (when open timing is set to DEFER or SCHEDULE) (1/3)

Utility	**No shutdown**		**Command shutdown**		**Ref-possible shutdown**	
	Open	**Closed**	**Open**	**Closed**	**Open**	**Closed**
Database structure modification utility	Y	Y	S	S	S	Y
Database load utility	Y	Y	Y	N	Y	Y
Database reorganization utility	Y	Y	Y	N	Y	Y

Utility	No shutdown		Command shutdown		Ref-possible shutdown	
	Open	Closed	Open	Closed	Open	Closed
Reflection processing for online reorganization (`pdorend` command)	Y	Y	S	S	S	Y

Legend:

Y: Can be set to check pending status.

S: Cannot be set to check pending status. The utility skips this processing and executes the next processing.

N: Cannot be set to check pending status.

Table C-13: Whether or not the check pending status can be set by a utility depending on the RDAREA's status (when open timing is set to DEFER or SCHEDULE) (2/3)

Utility	Reference-possible backup hold		Updatable backup-hold		Error shutdown	
	Open	Closed	Open	Closed	Open	Closed
Database structure modification utility	S	S	Y	Y	S	S
Database load utility	N	N	Y	Y	N	N
Database reorganization utility	N	N	Y	Y	N	N
Reflection processing for online reorganization (`pdorend` command)	S	S	Y	Y	S	S

Legend:

Y: Can be set to check pending status.

S: Cannot be set to check pending status. The utility skips this processing and executes the next processing.

N: Cannot be set to check pending status.

Table C-14: Whether or not the check pending status can be set by a utility depending on the RDAREA's status (when open timing is set to DEFER or SCHEDULE) (3/3)

Utility	No-log shutdown		Synchronization shutdown		Online reorganization hold	
	Open	Closed	Open	Closed	Open	Closed
Database structure modification utility	S	S	S	S	Y	Y
Database load utility	N	N	W*	W*	Y	Y
Database reorganization utility	N	N	W*	W*	Y	Y
Reflection processing for online reorganization (`pdorend` command)	S	S	S	S	Y	Y

Legend:

Y: Can be set to check pending status.

W: Waits until the shutdown status is released and then sets the check pending status.

S: Cannot be set to check pending status. The utility skips this processing and executes the next processing.

N: Cannot be set to check pending status.

* A utility or UAP is placed in lock-release wait status until the shutdown status is released. After the shutdown status is released, the utility or UAP can be executed.

D. Maximum Number of Concurrently Executable Utilities

Table D-1 shows the maximum number of utilities that can be executed concurrently. The table assumes that appropriate values are set in the system definition. For settings, see the *HiRDB Version 8 System Definition*.

- Maximum number of concurrent connections (system common definition: `pd_max_users`)
- Maximum number of processes that can be started per back-end server (server common definition, back-end server definition: `pd_max_bes_process`)

Table D-1: Maximum number of utilities that can be executed concurrently

Utility	**Maximum number of concurrently executable copies**		**Number of connections required for execution of 1 utility**
	pd_utl_exec_mode=0	**pd_utl_exec_mode=1**	
Database initialization utility	1		1
Database structure modification utility	1		1
Database load utility	32	Value specified in `pd_max_users` operand	1
Database reorganization utility[2]	32	Value specified in `pd_max_users` operand	1
Rebalancing utility	32	Value specified in `pd_max_users` operand	1
Database definition utility[3]	Value specified in `pd_max_users` operand		1
Integrity check utility	Value specified in `pd_max_users` operand ÷ 2		2
Database condition analysis utility	16	Value specified in `pd_max_users` operand	1
Optimizing information collection utility	16	Value specified in `pd_max_users` operand ÷ 2	2
Dictionary import/export utility	1		1

Utility	Maximum number of concurrently executable copies		Number of connections required for execution of 1 utility
	pd_utl_exec_mode=0	pd_utl_exec_mode=1	
Database copy utility	$32 \geqq \sum_{i=1}^{e} gi$ *gi*: Number of backup files specified in `pdcopy` *e*: Maximum number of copies of `pdcopy` that can be executed concurrently[1]	Value specified in pd_max_users operand $\geqq \sum_{i=1}^{e} gi$ *gi*: Number of backup files specified in `pdcopy` *e*: Maximum number of copies of `pdcopy` that can be executed concurrently[1]	0
Database recovery utility	32	Value specified in `pd_max_users` operand	0

[1] The maximum number of copies of a utility that can be executed concurrently also depends on the number of backup files specified in `pdcopy`. This can be determined as follows:

- If each copy of the `pdcopy` utility uses one backup file and if six such copies of the utility are executed concurrently, the number of concurrently executing copies will be 1 x 6 = 6.
- If each copy of the `pdcopy` utility uses two backup files and if four such copies of the utility are executed concurrently, the number of concurrently executing copies will be 2 x 4 = 8.
- If two such sets of copies of the utility are run simultaneously, the number of concurrently executing copies will be 6 + 8 = 14.

[2] When the free page release utility or the global buffer residence utility is used, the database reorganization utility is executed internally.

[3] Lock-release wait status occurs because definition SQL statements cannot be executed concurrently.

E. Creation of Input Data Files for the Database Load Utility

The following types of files can be input to the database load utility (`pdload`):

File type	Record length	DAT format	Binary format or fixed-size data format	Fixed-size data format
Regular file	Any	Y	Y	Y
FIFO	Any[1]	Y	Y	N
Fixed-length blocked tape	512-byte	Y[2]	N	N
Variable-length blocked tape	32KB(any length for last block)	Y	Y	Y
EasyMT tape	Any[3]	Y	Y	Y

Y: Can be input.

N: Cannot be input.

[1] The maximum record length is 32 KB in the normal DAT format, and is the value of the `maxreclen` operand in the `source` statement in the extended DAT format.

[2] The end of the tape is padded with 0x00 at a 512-byte boundary.

[3] The data length equals the record length, except for EasyMT.

Because the procedures for creating the following three types of files are complicated, this appendix explains the procedures:

- Fixed-length blocked tape
- Variable-length blocked tape
- EasyMT

E.1 Creating an input file on fixed-length blocked tape

(1) Principal media

`CMT,AT`

(2) File name examples

/dev/cmt00 (internal CMT), /dev/sysdat (internal DAT)

(3) Overview

The length of a fixed-length blocked tape is an integer multiple of 512-byte blocks. Because management information is not recorded on the medium, only 512-byte integer multiples of data can be handled. Therefore, this medium cannot handle binary format.

Files in DAT format are handled by padding the unused portion of the last block with 0x00 (0x00 is not normally found in DAT-format files).

(4) Creation examples

(a) Regular file to fixed-length blocked tape

Copy DAT-format file `'/etc/csh.login'` to CMT internal tape '/dev/cmt00':

```
dd if=/etc/csh.login of=/dev/cmt00 conv=sync

→ 5+1 input records
→ 6+0 output records
```

→ System output

Explanation:

5+1 indicates that five 512-byte blocks and one block with fewer than 512 bytes are to be input.

6+0 indicates that six 512-byte blocks are to be output (the last block is padded with 0x00).

E.2 Creating an input file on variable-length blocked tape

(1) Principal media

`DAT,CGMT,OMT`

(2) File name examples

`/dev/dat/vdat010` (internal DAT),

`/dev/cgmt/vcgmt160` (cartridge magnetic tape),

`/dev/omt/vomt055` (open reel magnetic tape)

(3) Creating a file name

To install a variable-length blocked device, the file name must first be created.

The super user then executes the following command:

```
/etc/mknod /dev/dat/vdat010   c 119 0x010001
/bin/chmod 0666 /dev/dat/vdat010
```

Explanation:

Normally, when this command is executed, a variable-length blocked access device for the internal DAT is created.

`/dev/dat/vdat010`: Device name

`c`: Character special file

`119`: Major number (indicates DAT variable-block usage)

`0x010001`: Minor number (indicates internal DAT will be used in compressed mode)

See man below for unknown words or concepts.

mknod or chmod: man lm mknod or man chmod

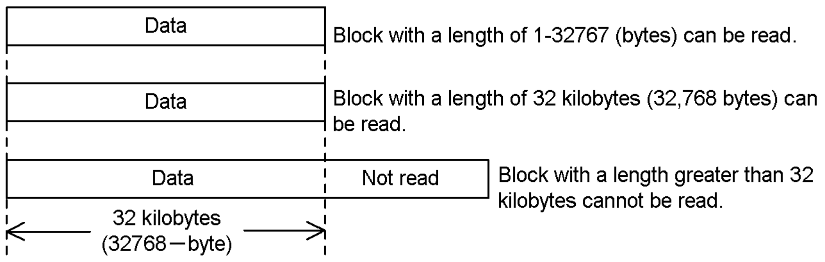
Instruction name rule for major numbers, minor numbers, and device names: man 6 mt

(4) Overview

Blocks of any lengths can be written on variable-length blocked tape.

Because blocks of any lengths can be written, this type of tape can directly handle data in binary format, in addition to data in DAT format. The tape must be read in blocks the same length as they were written (for example, if a block written in 64 KB is read in 32 KB, the remaining 32 KB are discarded). Because there is no means of communicating to a reader that the record length changes, all blocks except the last block are normally written at the same length, and only the last block is written as a fractional block length.

Because the database load utility (`pdload`) makes that last block 32 KB, tapes with block lengths longer than 32 KB cannot be read correctly.


Data
Block with a length of 1-32767 (bytes) can be read.
Data
Block with a length of 32 kilobytes (32,768 bytes) can be read.
Data
Not read
Block with a length greater than 32 kilobytes cannot be read.
32 kilobytes (32768−byte)

When a file is created, the block length should be 32 KB or less.

In the following example, the block length is set at 32 KB when the file is created.

(5) Creation example

The file is created in the same way in DAT format and binary format.

(a) Regular file to variable-length blocked tape

Because the `cp` command cannot set the buffer length, it cannot be used; the `dd` command must be used instead.

/usr/bin/vi output to /dev/dat/vdat010 variable-length blocked tape

```
dd if=/usr/bin/vi of=/dev/dat/vdat010 bs=32k

→  7+1 input records
→  7+1 output records
```

→ System output

Explanation:

7+1 indicates that seven 32-KB blocks and one block with fewer than 32 KB were processed.

(b) Creating programs

When the following types of programs are created, data of any length (specified by data_len) is output to variable-length blocked tape in 32-KB blocks. This is done by allocating the 32-KB write_buf to the system buffer and specifying full buffering (_IOFBF).

```
#define PROC_BUFSIZ     1024 * 32
FILE*   fp;
int     data_len;               /* data length (needed in binary format) */
char    write_buf[PROC_BUFSIZ]; /* buf system call */
char    data_buf[PROC_BUFSIZ];  /* buf user */
    :
fp = fopen("/dev/dat/vdat010","w")
setvbuf(fp, write_buf, _IOFBF, PROC_BUFSIZ)
    :
while(while there is data) {
    data write processing
}
    :
fclose(fp);
```

Explanation:

Data write processing

- DAT format

  Load data into data_buf. Data + \n + \0

  fputs (data_buf, fp);

- Binary format

  Load data into data_buf. The data length is data_len.

  fwrite (data_buf, data_len, 1, fp);

(c) Data loaded mistakenly with the cp command

If the original data was in a regular file, it must be reloaded using the method in (a).

When there are two DAT devices, the buffer length can be converted with the following method:

```
dd if=/dev/vdat010 ibs=64k of=/dev/vdat011 obs=32k
```

Explanation:

`/dev/vdat010`: Data loaded mistakenly with the `cp` command

`/dev/vdat011`: Tape to be newly loaded for pdload

E.3 Creating an input file on EasyMT tape

EasyMT is a program product that uses a variable-length blocked tape such as in E.2 Creating an input file on variable-length blocked tape and provides tape management by means of volume names and file names.

(1) Main media

`DAT,CGMT,OMT`

(2) File name examples

/dev/dat/vdat010 (internal DAT),

/dev/cgmt/vcgmt160 (cartridge magnetic tape),

/dev/omt/vomt055 (open reel magnetic tape)

(3) Creating a file name

To install a variable-length blocked device, the file name must first be created.

The super user then executes the following command:

```
/etc/mknod /dev/dat/vdat010   c 119 0x010001
/bin/chmod 0666 /dev/dat/vdat010
```

Explanation:

Normally, when this command is executed, a variable-length blocked access device for the internal DAT is created.

`/dev/dat/vdat010`: Device name

`c`: Character special file

`119`: Major number (indicates DAT variable block usage)

`0x010001`: Minor number (indicates internal DAT is used in compressed mode)

Refer to man below for unknown words or concepts.

mknod or chmod: man lm mknod or man chmod

Instruction name rule for major numbers, minor numbers, and device names: man 6 mt

(4) Overview

EasyMT has the following header formats:

Symbol	Description
`EMTVOL_EL`	EasyMT label
`EMTVOL_SL`	Standard label
`EMTVOL_JL`	JIS code label
`EMTVOL_NL`	No label

The header is read first by the database load utility (`pdload`), and then items in a recognizable format are read. An error results if the format is not recognizable or there is no label.

EasyMT has the following record formats:

Symbol	Description	Block specification	Record specification	Label specification
`F`	Fixed-length non-blocked records	N	F	SL or EL
`FB`	Fixed-length blocked records	B		
`V`	Variable-length non-blocked records	N	V	
`VB`	Variable-length blocked records	B		
`U`	Unspecified-length records	N	U	

The database load utility (`pdload`) can support all the above formats. However, the maximum of the set containing the block length, record length, or data length is as follows:

Model where file is created	Maximum value
3050RX Group	32KB (32768-BYTE)
3500 Series	
HITAC-M Series	32KB-8 (32760-BYTE)

Because the length of one record in the F and FB record formats is fixed, only FIX tables and tables in which the column structures of all rows are fixed-length and which are input as a binary file can use these formats. If the input is an ASCII file, these formats can be used only when all data lengths are the same.

The database load utility (pdload) can handle the following two types of files as input files:

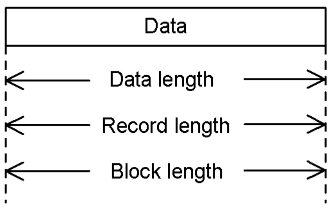
- Binary-format files (-b specified)

- DAT-format files (-b not specified (default))

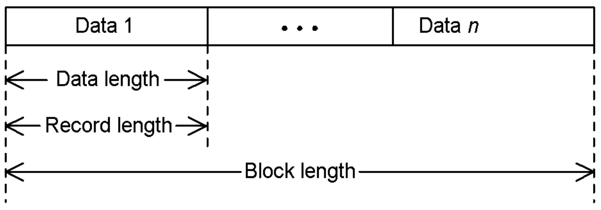
(a) Binary-format files

Binary-format files have the record formats shown as follows.

- **F (Fixed-length non-blocked records)**



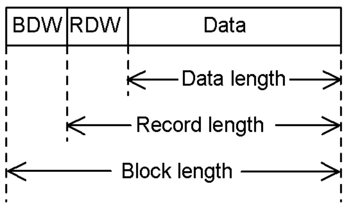
- **FB (Fixed-length blocked records)**



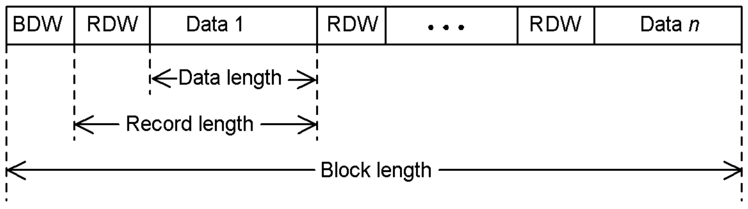
One row is specified as the data length (record length).

F or FB can be specified only for FIX tables or tables in which the column structures of all rows are fixed in length.

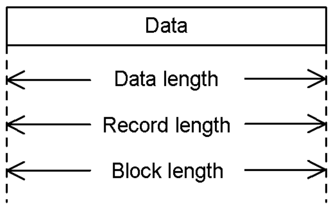
- **V (Variable-length non-blocked records)**



- **VB (Variable-length blocked records)**



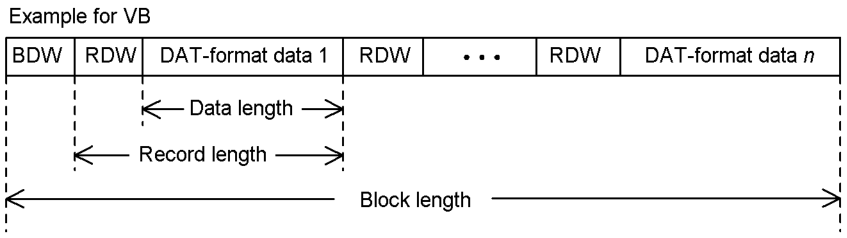
- **U (Unspecified-length records)**



(b) DAT-format files

DAT-format files have one record per row (up to NL characters). The normal record formats are V, VB, or U. The F and FB formats can be used only if the data lengths are all the same.

The data length includes NL (LF: `0x0a`, `'\n'`). Records that do not include NL result in an error.



The maximum and minimum values for the data length, record length, and block length above are as follows:

File created on the 3050RX Group and 3500 Series

Record Format	Data length		Record length		Block length	
	min	max	min	max	min	max
F/FB	1	32768	1	32768	1	32768
V/VB	1	32760	5	32764	9	32768
U	1	32768	—	—	1	32768

File created on the HITAC-M Series

Record Format	Data length		Record length		Block length	
	min	max	min	max	min	max
F/FB	1	32760	1	32760	1	32760

Record Format	Data length		Record length		Block length	
	min	max	min	max	min	max
V/VB	1	32752	5	32756	9	32760
U	1	32760	—	—	1	32760

(5) Input file creation example

Following is an example of creating an input file on the 3050RX Group or the 3500 Series for the database load utility (`pdload`) using the functions provided by EasyMT.

The EasyMT definition is specified in the MT attributes definition file.

(a) Program example

The MT attributes definition file is specified in the first argument (`argv[1]`).

Specify error processing as required.

```
#include    <stdio.h>
#include    <stdlib.h>
#include    <unistd.h>
#include    "easymt.h"
#define     PROC_BUFSIZ     1024 * 32
char        buf[PROC_BUFSIZ];
int         data_len;               /* data length */
    :
void main(int argc, char *argv[])
{

    EmtDescr_t* emt_fp;
    if (EmtInit(NULL, argv[1])) {
        printf("EmtInit failure, ,EmtErrno=%i\n", EmtErrno);
        exit(EXIT_FAILURE);
    }
    if (emt_fp = EmtAlloc(NULL, argv[1])) {
    } else {
        printf("EmtAlloc failure, ,EmtErrno=%i\n", EmtErrno);
        exit(EXIT_FAILURE);
    }

    if (EmtFopen(emt_fp, NULL)) {
        printf("EmtFopen failure, ,EmtErrno=%i\n", EmtErrno);
        goto proc_free;
    }
    while(while there is data) {
            :

Data is loaded into buf; the data length is assigned to data_len.
For DAT format data, NL (0x0a) is included in the data.

            :

        rc = EmtWrite(emt_fp, buf, data_len);
        if (rc != data_len) {
            printf("EmtWrite failure,EmtErrno=%i\n", EmtErrno);
            goto proc_close;
        }
    }
proc_close:
    if (EmtFclose(emt_fp)) {
        printf("EmtFclose failure,EmtErrno=%i\n", EmtErrno);
        goto proc_free;
    }


proc_free:
    if (EmtFree(emt_fp)) {
        printf("EmtFree failure,EmtErrno=%i\n", EmtErrno);
        exit(EXIT_FAILURE);
    }
}
```

(b) MT attributes definition file example

In the definition file, a blank immediately following an equals sign (=) means that the corresponding value is omitted.

Device name: /dev/dat/vdat010

Buffer sectors count: 2

Volume name: HiRDB

Label type: Standard label (SL)

File name: input_data

Open mode: Write-specified

Record type: VB maximum value

To handle the M-series, the maximum value is set to 32 KB - 8 bytes (32,760 bytes).

A data length of up to the record length - 4 bytes (32,752 bytes) can be handled.

The following entries in the MT attributes definition file correspond to pdload control information file source statement entries:

MT Attributes definition file entry	pdload control information file source statement entry	Maximum number of characters
volname=	vol=	6
filename=	file=	17*

* When an MT file is created on the HITAC-M series, a file name of up to 44 characters can be specified; however, only the last 17 characters are valid characters, and must be specified as such.

Because the bufno entry is simply the specification of the buffer sectors count to be used for writing, there is no need to match it to the pdload specification.

An example definition for creating the above file is as follows:

```
#/*-----------------------------------------------------------
#
(1) Operating specifications
#/*-----------------------------------------------------------
devname=/dev/dat/vdat010
blking=OFF
bufno=2
#/*-----------------------------------------------------------
#   (2) Volume attribute specifications
#/*-----------------------------------------------------------
volname=HiRDB
voltype=SL
#/*-----------------------------------------------------------
#   (3) File attribute specifications
#/*-----------------------------------------------------------
filename=input_data
openmode=EMT_WRONLY
blklen=32760
reclen=32756
blktype=B
rectype=V
```

F. Creation of a UOC for Use by pdload and pdrorg

F.1 Creating a shared library

After you finish coding the UOC, you must create a shared library. Specify the created shared library and the functions to be called in the control statements for `pdload` or `pdrorg`.

(1) For HP-UX

This example creates a shared library named `libuoc.sl` from the three source files `sample1.c`, `sample2.c`, and `sample3.c`:

1. Compile the files with the `+z` option specified.

```
$ /usr/bin/cc -c -I $PDDIR/include +z sample1.c sample2.c
sample3.c
sample1.c:
sample2.c:
sample3.c:
```

2. Link the created files with extension `.o`.

```
$ /bin/ld -b -o libuoc.sl sample1.o sample2.o sample3.o
```

(a) For HiRDB in the 64-bit mode

To execute the UOC on a HiRDB in the 64-bit mode, you must compile it in the 64-bit mode. In this case, specify `+DD64` as the compile option. For the HP-UX (IPF) version, specify `.so` as the extension for shared libraries.

(b) POSIX library version of HiRDB

To execute the UOC on a POSIX library version of HiRDB, multi-threads must be supported. Note that multi-threads cannot be used within the UOC.

If you are using the POSIX library version of HiRDB, also note the following:

- You must use HP-UX 11.0.
- To support multi-threads, specify the following options during compilation:

  ```
  -D_REENTRANT -D_POSIX_C_SOURCE=199506L -D_THREAD_SAFE
  -D_HPUX
  ```
- Use thread-safe functions.

(2) For Solaris

This example creates a shared library named `libuoc.so` from the three source files `sample1.c`, `sample2.c`, and `sample3.c`:

1. Compile the files.

```
$ /usr/ucb/cc -c -I$PDDIR/include sample1.c sample2.c
sample3.c
sample1.c:
sample2.c:
sample3.c:
```

2. Link the created files with extension `.o`.

```
$ /usr/css/bin/ld -G -o libuoc.so sample1.o sample2.o
sample3.o
```

(a) HiRDB in the 64-bit mode

To execute the UOC on a HiRDB in the 64-bit mode, you must compile it in the 64-bit mode. In this case, specify `-xarch=v9` as the compile option.

(3) For Linux

This example creates a shared library named `libuoc.so` from the three source files `sample1.c`, `sample2.c`, and `sample3.c`:

1. Compile the files.

```
$ /usr/bin/cc -c -I$PDDIR/include sample1.c sample2.c
sample3.c
sample1.c:
sample2.c:
sample3.c:
```

2. Link the created files with extension `.o`.

```
$ /usr/bin/cc -shared -o libuoc.so sample1.o sample2.o
sample3.o
```

(4) For AIX 5L

This example creates a shared library named `libuoc.so` from the three source files `sample1.c`, `sample2.c`, and `sample3.c`:

1. Compile the files.

```
$ /usr/vac/bin/xlc -c -I$PDDIR/include sample1.c sample2.c
sample3.c
sample1.c:
sample2.c:
sample3.c:
```

2. Link the created files with extension `.o`.

```
$ /usr/vac/bin/xlc -G -o libuoc.so sample1.o sample2.o
sample3.o
```

(a) HiRDB in the 64-bit mode

To execute the UOC on a HiRDB in the 64-bit mode, you must compile it in the 64-bit mode. In this case, specify `-q64` as the compile option.

(b) POSIX library version of HiRDB

To execute the UOC on a POSIX library version of HiRDB, multi-threads must be supported. Note that multi-threads cannot be used within the UOC.

If you are using a POSIX library version of HiRDB, also note the following:

During compilation, use the `xlc_r` command.

For details about supporting multi-threads, see the applicable OS documentation.

F.2 Notes on UOC creation

When a UOC is used, a user-created program is installed on HiRDB for processing. In other words, the user-created program becomes part of the database management system. Any problem with the UOC may develop into a problem with HiRDB itself. For this reason, the UOC must be created very carefully and tested thoroughly. In addition, UOC creation must follow the HiRDB programming rules.

(1) Common rules for pdload and pdrorg

1. The programming language that can be used is C language.
2. Create the UOC in the shared libraryformat.
3. Grant read and execution permissions to the created shared library. Do not grant write permission. If you grant write permission by mistake, the UOC execution time becomes long, resulting in adverse effects on performance.
4. A function that accesses a database uses a global buffer. A function that accesses a database from a buffer also uses a global buffer. If the buffer usage time is long, HiRDB detects a system error. Therefore, minimize the UOC processing and return control back to `pdload` or `pdrorg` as soon as possible.

5. If processing terminates abnormally within the UOC, `pdload` or `pdrorg` also terminates abnormally. For details about the database status in the event of abnormal termination and the recovery method, see *5.12 Database status in the event of an error and recovery methods* or *8.13 Database status in the event of an error and recovery method*.
6. The header file for UOC creation (`pdutluoc.h`) that declares the UOC interface area and symbolic constants is located under `$PDDIR/include`. To reference or update the interface area, we recommend that you include this header file during UOC compilation. If you are defining a user-specific header file for UOC creation for reasons such as variable name settings, note that there are differences in the table structure.
7. Make sure that no global variable or function name begins with any of the following characters:
    - Uppercase `SQL`, `Y`, or `Z`
    - Lowercase `p_`, `pd`, `yy`, or `z`

    If you are using plug-ins or Java stored routines, also make sure that none of the names begin with any of the following characters:

    When using plug-ins:

    Lowercase `_p`

    When using Java stored routines:

    Lowercase `da`

    Lowercase `dbr`

    Lowercase `dp`
8. Use the `void` type for the UOC's return value.
9. Do not use the UOC to set or change environment variables.
10. No SQL statement can be issued within the UOC.
11. Do not perform signal manipulation.
12. The `main` function cannot be used as the UOC.
13. Do not create a recursive function.
14. A stack shortage may result in abnormal termination within the UOC. If you use a large amount of stack, change the value of the relevant operating system parameter. For details about the operating system parameters, see the *HiRDB Version 8 Installation and Design Guide*.
15. Do not create threads.

16. Do not use any functions other than the following:
    - File manipulation functions
    - Character processing functions
    - Memory allocation and release functions
17. Do not use any of the following functions:
    - Process manipulation functions, such as `fork()`, `exit()`, `abort()`, and `exec()`
    - `sleep()`, `select()`, and `wait()`
    - Stack manipulation functions (such as `setjmp()` and `longjmp()`)
    - Shared memory manipulation functions
    - Semaphore manipulation functions
    - Socket manipulation functions
    - System resources manipulation functions (such as `setrlimit`)
    - `mmap()` and `munmap()`
    - `gethostent()`, `sethostent()`, `endhostent()`, `gethostbyname()`, `gethostbyaddr()`, and `herror()`
    - `tempnam()` and `tmpnam()`
    - `pstat()`
    - `system()`
18. Do not use within the UOC the ID of a process that is executed by the UOC.
19. With respect to file manipulation, note the following:
    - Do not manipulate any file in the HiRDB installation directory or HiRDB directory.
    - Do not manipulate any OS file.
    - To create a file, specify its absolute path name. Otherwise, a file will be created in the current directory. The current directory of a HiRDB process is under `$PDDIR/tmp/home`; do not create files here.
    - If a file is to be created within a UOC, the file permissions must be set explicitly.
    - Make sure that all temporary files created by the UOC are deleted (unlinked).
    - Do not use any special files, such as `PIPE`.
    - Do not use the standard input, standard output, or standard error output.

- If you execute multiple utilities, multiple sets of the UOC are also executed concurrently. If the UOC locks file resources, make sure that any of the other UOCs that are running concurrently will not be placed in lock-release wait status.
- The user is responsible for managing files output by the UOC.

(2) pdload-specific rules

1. For a HiRDB/Parallel Server, provide the shared librarycontaining the UOC at the host where the server specified in the `source` statement is located.
2. If `pdload` is to read a binary-format input data file and pass a line of data to the UOC, but it cannot edit data to a single line due to invalid length of a variable-length character string, `pdload` cancels processing.
3. If there is no response from `pdload`, you can use the `pdls -d rpc` command to determine whether `pdload` or the UOC has control. If the resulting `USR_EVENT` is `0x00052601`, the UOC has control.

(3) pdrorg-specific rules

1. Provide the shared librarycontaining the UOC at the host where the server used to call the UOC is located. For details about the server that calls the UOC and the host, see *8.10.2 Relationships between options and control statements*.
2. The UOC must not manipulate any files other than UOC data files.
3. If memory space is allocated by the UOC by means of a OS functions (such as `malloc`), make sure that the memory space is released by the following call methods:
   - Returning a nonzero value as the return code during startup processing
   - Termination processing
   - Cancellation processing
4. If there is no response from `pdrorg`, you can use the `pdls -d rpc` command to determine whether `pdrorg` or the UOC has control. If the resulting `USR_EVENT` is `0x00051601`, the UOC has control.

G. Number of Concurrent Command Connections

Some commands (operation commands and utilities) perform connection processing internally in HiRDB. When you execute commands or determine the `pd_max_users` operand value in the system definition, you must note the following:

The connection frame specified in the `pd_max_users` operand is used at the time of command connection; therefore, the number of connections available to users is reduced temporarily.

If there are more command connections than the number of connections available in the connection frame at the time of a command execution, a connection error occurs and the command may result in an error.

Table G-1 shows the number of additional concurrent connections that are possible for each command. The provided number of concurrent connections is relative to there already being one command execution.

Table G-1: Number of additional concurrent connections for each command

Command name	Number of additional concurrent connections
`pdacunlck`	0
`pdadmvr`	0
`pdaudbegin`	0
`pdaudend`	0
`pdaudrm`	0
`pdaudswap`	1
`pdbkupls`	0
`pdbufls`	0
`pdbufmod`	0
`pdcancel`	0
`pdcat`	0
`pdchgconf`	0
`pdchprc`	0
`pdclose`	0

Command name	Number of additional concurrent connections
pdclttrc	0
pdcmt	0
pdconfchk	0
pdconstck	2
pdcopy	0
pdcspool	0
pddbadset	0
pddbchg	0
pddbfrz	0
pddbls	0
pddbst	1
pddef	1
pddefrev	1
pdexp	1
pdfbkup	0
pdffsck	0
pdfgt	0
pdfls	0
pdfmkfs	0
pdfrm	0
pdfrstr	0
pdfstatfs	0
pdgen	0
pdgetcst	2
pdgeter	0
pdgrprfl	0

Command name	Number of additional concurrent connections
`pdhold`	0
`pdinit`	1
`pditvstop`	0
`pditvtrc`	0
`pdjarsync`	0
`pdlistls`	0
`pdload`	1
`pdlodsv`	0
`pdlogadpf`	0
`pdlogatul`	0
`pdlogchg`	0
`pdlogcls`	0
`pdloginit`	0
`pdlogls`	0
`pdlogopen`	0
`pdlogrm`	0
`pdlogswap`	0
`pdlogsync`	0
`pdlogucat`	0
`pdlogunld`	0
`pdls -d aud`	When HiRDB is active, 1; when HiRDB is inactive, 0
`pdls` (other than `-d aud`)	0
`pdmemsv`	0
`pdmod`	1
`pdobils`	0
`pdobjconv`	1

Command name	Number of additional concurrent connections
`pdopen`	0
`pdopsetup`	0
`pdorbegin`	0
`pdorcheck`	1
`pdorchg`	0
`pdorcreate`	1
`pdorend`	(number of servers specified in the `pdorend -s` option) × (value of the `pdorend -m` option)
`pdpfresh`	0
`pdpgbfon`	1
`pdplgrgst`	1
`pdplgset`	0
`pdprgcopy`	0
`pdprgrenew`	0
`pdrbal`	1
`pdrbk`	0
`pdrdrefls`	1
`pdreclaim`	1
`pdreginit`	1
`pdrels`	0
`pdrorg`	1
`pdrisechk`	For a HiRDB/Single Server, 1; for a HiRDB/Parallel Server, the number of units that contain a dictionary server or back-end server
`pdrisedbto`	0
`pdriseset`	0
`pdrpause`	0
`pdrplstart`	0

Command name	Number of additional concurrent connections
pdrplstop	0
pdrstr	0
pdsetup	0
pdsql	1
pdstart	0
pdstbegin	0
pdstedit	0
pdstend	0
pdstjswap	0
pdstjsync	0
pdstop	0
pdstscls	0
pdstsinit	0
pdstsopen	0
pdstsrm	0
pdstsswap	0
pdsvhostname	0
pdtrndec	0
pdtrnqing	0
pdusrchk	0
pdvrup	1
pdvwopt	0

H. List of Command Return Codes

Table H-1 lists and describes the return codes that can be set by HiRDB commands.

Table H-1: List of return codes set by commands

Command	Output message		Return code	Description
	Message ID	Code		
`pdacunlck`	`KFPD01113-I`	0	0	Normal termination
		8	8	Abnormal termination
`pdadmvr`	--	--	0	Normal termination
			8	Abnormal termination (invalid option)
`pdaudbegin`	--	--	0	Normal termination
			4	Some units terminated normally.
			8	Abnormal termination
`pdaudend`	--	--	0	Normal termination
			4	Some units terminated normally.
			8	Abnormal termination
`pdaudrm`	--	--	0	Normal termination
			8	Abnormal termination
`pdaudswap`	--	--	0	Normal termination
			8	Abnormal termination
`pdbkupls`	`KFPR26276-I`	0	0	Normal termination
		12	12	Abnormal termination
`pdbufls`	--	--	0	Normal termination
			4	Warning termination (some server processes terminated with error)
			8	Abnormal termination
			12	Abnormal termination (an event disabling display of error messages occurred)

Command	Output message		Return code	Description
	Message ID	Code		
pdbufmod	—	—	0	Normal termination
			4	Warning termination (some server processes terminated with error)
			8	Abnormal termination
			12	Abnormal termination (an event disabling display of error messages occurred)
pdcancel	—	—	0	Normal termination
			8	Abnormal termination (such as invalid option or rsh error)
pdcat	—	—	0	Normal termination
			8	Abnormal termination
pdcbl	—	—	0	Normal termination
			4	Error occurred (process was resumed through the end).
			8	Error occurred (process was resumed through the end).
			12	Error occurred (process was cancelled).
			16	Error occurred (process was cancelled).
pdchgconf	KFPS04661-I	0	0	Normal termination
		8	8	Abnormal termination
pdchprc	—	—	0	Normal termination
			8	Abnormal termination (such as invalid option or rsh error)

Command	Output message		Return code	Description
	Message ID	Code		
pdclose	—	—	0	Normal termination
			4	Warning termination (some RDAREA processes terminated with error)
			8	Abnormal termination
			12	Abnormal termination (an event disabling display of error messages occurred)
pdclttrc	—	—	0	Normal termination
			8	Abnormal termination
pdcmt	—	—	0	Normal termination
			1	Abnormal termination (such as invalid option or `rsh` error)
pdconfchk	KFPS05007-I	0	0	Normal termination
		8	8	Abnormal termination (invalid argument or `pdconfchk` command execution error)
pdconstck	KFPL50001-I	0	0	Normal termination
		4	4	Normal termination (constraint violation occurred) This includes when processing was cancelled because the number of key values resulting in a constraint violation exceeded the maximum number permissible.
		8	8	Abnormal termination
pdcpp	—	—	0	Normal termination
			4	Error occurred (process was resumed through the end).
			8	Error occurred (process was resumed through the end).
			12	Error occurred (process was cancelled).

Command	Output message		Return code	Description
	Message ID	Code		
			16	Error occurred (process was cancelled).
pdcopy	KFPR00756-I	0	0	Normal termination
		8	8	Abnormal termination (some copy processes failed or were skipped)
		12	12	Abnormal termination (all copy processes failed)
pdcspool	—	—	0	Normal termination
			4	Normal termination
			12	Abnormal termination
pddbadset	KFPS04631-I	0	0	Normal termination
		8	8	Abnormal termination
pddbchg	—	—	0	Normal termination
			4	Warning termination (some RDAREA processes terminated with error)
			8	Abnormal termination
			12	Abnormal termination (an event disabling display of error messages occurred)
pddbfrz	—	—	0	Normal termination
			4	Warning termination (some RDAREA processes terminated with error)
			8	Abnormal termination
			12	Abnormal termination (an event disabling display of error messages occurred)

Command	Output message		Return code	Description
	Message ID	Code		
`pddbls`	—	—	0	Normal termination
			4	Warning termination
			8	Abnormal termination
			12	Abnormal termination (an event disabling display of error messages occurred)
`pddbst`	`KFPK10301-I`	0	0	Normal termination
		4	4	Abnormal termination (invalid specification)
		8	8	Abnormal termination
`pddef`	—	—	0	Normal termination
			8	Abnormal termination
`pddefrev`	`KFPX18400-I`	0	0	Normal termination
		8	8	Abnormal termination
		12	12	Abnormal termination
`pdexp`	`KFPX18400-I`	0	0	Normal termination
		4	4	Normal termination (invalid procedure or trigger export)
		8	8	Abnormal termination
		12	12	Abnormal termination
`pdfbkup`	—	—	0	Normal termination
			1	Warning termination
			-1	Abnormal termination
`pdffsck`	—	—	0	Normal termination
			1	Warning termination (integrity error was detected during verification)
			-1	Abnormal termination

Command	Output message		Return code	Description
	Message ID	Code		
pdfgt	—	—	0	Normal termination
			1	Abnormal termination (such as invalid option or rsh error)
pdfls	—	—	0	Normal termination
			-1	Abnormal termination
pdfmkfs	—	—	0	Normal termination
			-1	Abnormal termination
pdfrm	—	—	0	Normal termination
			-1	Abnormal termination
pdfrstr	—	—	0	Normal termination
			1	Warning termination
			-1	Abnormal termination
pdfstatfs	—	—	0	Normal termination
			-1	Abnormal termination
pdgen	—	—	0	Normal termination
pdgetcst	KFPN00011-I	0	0	Normal termination
		4	4	Normal termination (warning occurred on data dictionary table manipulation)
		8	8	Abnormal termination
pdgeter	KFPN10301-I	0	0	Normal termination
		4	4	Warning termination (part of the information acquisition processing was skipped)
		8	8	Abnormal termination
		12	12	Termination by interrupt
pdgrprfl	KFPD01102-I	0	0	Normal termination
		8	8	Abnormal termination

Command	Output message		Return code	Description
	Message ID	Code		
pdhold	—	—	0	Normal termination
			4	Warning termination (some RDAREA processes terminated with error)
			8	Abnormal termination
			12	Abnormal termination (an event disabling display of error messages occurred)
pdinit	KFPX24013-I	0	0	Normal termination
		4	4	Normal termination (warning-level error occurred, but processing terminated normally)
		8	8	Normal termination (initialization terminated normally, but communication for the termination of initialization resulted in an error)
		12	12	Abnormal termination
pditvstop	—	—	0	Normal termination
			1	Abnormal termination
pditvtrc	—	—	0	Normal termination
			1	Abnormal termination
pdjarsync	—	—	0	Normal termination
			4	Abnormal termination (JAR file has or has not been registered)
			8	Abnormal termination
pdlistls	—	—	0	Normal termination
			8	Abnormal termination
pdload	KFPL00704-I	0	0	Normal termination
		4	4	Normal termination (detection of input data error)
		8	8	Abnormal termination

Command	Output message		Return code	Description
	Message ID	Code		
`pdlodsv`	—	—	`0`	Normal termination
			`1`	Abnormal termination (user privilege error, etc.)
			`8`	Abnormal termination (environment variable has not been set)
			`12`	Abnormal termination (invalid option, user privilege error, etc.)
`pdlogadpf`	—	—	`0`	Normal termination
			`4`	Abnormal termination
			`8`	Abnormal termination (such as invalid option or `rsh` error)
`pdlogatul`	—	—	`0`	Normal termination
			`4`	Cancelled or timeout occurred.
			`8`	Abnormal termination (such as invalid option or `rsh` error)
`pdlogchg`	—	—	`0`	Normal termination
			`4`	Abnormal termination
			`8`	Abnormal termination (such as invalid option or `rsh` error)
			`12`	Abnormal termination (when processing was retried at the standby system in a configuration without inheritance of IP addresses)
`pdlogcls`	—	—	`0`	Normal termination
			`4`	Abnormal termination
			`8`	Abnormal termination (such as invalid option or `rsh` error)

Command	Output message		Return code	Description
	Message ID	Code		
`pdloginit`	—	—	`0`	Normal termination
			`4`	Abnormal termination
			`8`	Abnormal termination (such as invalid option or `rsh` error)
			`12`	Abnormal termination (when processing was retried at the standby system in a configuration without inheritance of IP addresses)
`pdlogls`	—	—	`0`	Normal termination
			`4`	Abnormal termination
			`8`	Abnormal termination (such as invalid option or `rsh` error)
			`12`	Abnormal termination (when processing was retried at the standby system in a configuration without inheritance of IP addresses)
`pdlogopen`	—	—	`0`	Normal termination
			`4`	Abnormal termination
			`8`	Abnormal termination (such as invalid option or `rsh` error)
`pdlogrm`	—	—	`0`	Normal termination
			`4`	Abnormal termination
			`8`	Abnormal termination (such as invalid option or `rsh` error)
			`12`	Abnormal termination (when processing was retried at the standby system in a configuration without inheritance of IP addresses)

Command	Output message		Return code	Description
	Message ID	Code		
`pdlogswap`	—	—	0	Normal termination
			4	Swapping occurred, but the synchronization point dump was not validated and the command was terminated forcibly.
			8	Abnormal termination (such as invalid option or `rsh` error)
`pdlogsync`	—	—	0	Normal termination
			4	Abnormal termination
			8	Abnormal termination (such as invalid option or `rsh` error)
`pdlogucat`	—	—	0	Normal termination
			8	Abnormal termination
`pdlogunld`	—	—	0	Normal termination
			4	Abnormal termination
			8	Abnormal termination (such as invalid option or `rsh` error)
			12	Abnormal termination (when processing was retried at the standby system in a configuration without inheritance of IP addresses)
`pdls(-d act)`	—	—	0	Normal termination
			8	Abnormal termination
`pdls(-d aud)`	—	—	0	Normal termination
			8	Abnormal termination
`pdls(-d ha)`	—	—	0	Normal termination
			8	Abnormal termination
`pdls(-d lck)`	—	—	0	Normal termination
			8	Abnormal termination

Command	Output message		Return code	Description
	Message ID	Code		
`pdls(-d mem)`	—	—	0	Normal termination
			8	Abnormal termination
`pdls(-d org)`	—	—	0	Normal termination
			8	Abnormal termination
`pdls(-d prc)`	—	—	0	Normal termination
			8	Abnormal termination
`pdls(-d ris)`	—	—	0	Normal termination
			8	Abnormal termination
`pdls(-d rpc)`	—	—	0	Normal termination
			8	Abnormal termination
`pdls(-d rpl)`	—	—	0	Normal termination
			8	Abnormal termination
`pdls(-d scd)`	—	—	0	Normal termination
			8	Abnormal termination
`pdls(-d stj)`	—	—	0	Normal termination
			8	Abnormal termination
`pdls(-d sts)`	—	—	0	Normal termination
			8	Abnormal termination
`pdls(d svr)`	—	—	0	All units are active and HiRDB startup processing was completed (after the `KFPS05210-I` message was displayed). `0` is also returned when all front-end servers were terminated by the `pdstop -s` command after HiRDB started.

Command	Output message		Return code	Description
	Message ID	Code		
			4	Some units are inactive
			4	Before completion of HiRDB startup processing (status of front-end server is `SUSPEND`)
			8	Abnormal termination
`pdls(-d ust)`	—	—	0	Unit is active (front-end server can be connected if it is located at the unit where the command is executed).
			4	Unit is starting (front-end server is located at the unit where the command is executed, but the front-end server cannot be connected), or unit is stopping.
			8	Restart of the process service was cancelled.
			12	Unit is stopped (`pdsetup -d` command can be executed).
			16	OS registration has been deleted.
`pdls(-d trn)`	—	—	0	Normal termination
			8	Abnormal termination
`pdmemsv`	`Memory saving level:`	—	0	Normal termination
	—	—	2	Abnormal termination (such as invalid option)
`pdmod`	`KFPX24213-I`	0	0	Normal termination
		4	4	Normal termination (warning-level error occurred, but processing terminated normally)
		8	8	Normal termination (some configuration change processes terminated normally)
		12	12	Abnormal termination
		16	16	Abnormal termination (inconsistent database)

Command	Output message		Return code	Description
	Message ID	Code		
pdobils	—	—	0	Normal termination
			4	Abnormal termination
pdobjconv	KFPX21002-I	0	0	Normal termination
		4	4	Warning termination
		8	8	Terminated normally, but some resulted in a re-registration error. Or, error occurred on command execution.
		12	12	Abnormal termination
pdocb	—	—	0	Normal termination
			4	Error occurred (process was resumed through the end).
			8	Error occurred (process was resumed through the end).
			12	Error occurred (process was cancelled).
			16	Error occurred (process was cancelled).
pdocc	—	—	0	Normal termination
			4	Error occurred (process was resumed through the end).
			8	Error occurred (process was resumed through the end).
			12	Error occurred (process was cancelled).
			16	Error occurred (process was cancelled).

Command	Output message		Return code	Description
	Message ID	Code		
pdopen	—	—	0	Normal termination
			4	Warning termination (some RDAREA processes terminated with error)
			8	Abnormal termination
			12	Abnormal termination (an event disabling display of error messages occurred)
pdopsetup	KFPS04618-I	—	0	Normal termination
	—	—	8	Privilege error, HiRDB is active, insufficient disk space, uninstalled, or already installed
			12	Invalid parameter or path
pdorbegin	KFPH27045-I	0	0	Normal termination
		4	4	Warning termination (processing was successful at some servers)
		8	8	Abnormal termination
		12	12	Abnormal termination (an event disabling display of error messages occurred)
pdorcheck	—	—	0	Normal termination
			4	Warning termination (some resources do not satisfy the conditions)
			8	Abnormal termination
pdorchg	KFPH27045-I	0	0	Normal termination
		4	4	Warning termination (processing was successful at some servers)
		8	8	Abnormal termination
		12	12	Abnormal termination (an event disabling display of error messages occurred)

Command	Output message		Return code	Description
	Message ID	Code		
pdorcreate	KFPT02014-I	0	0	Normal termination
		4	4	Warning termination (requested processing has already been executed, or deletion processing was cancelled by interactive entry)
		8	8	Abnormal termination
pdorend	KFPH27045-I	0	0	Normal termination
		4	4	Warning termination for one of the following reasons: • The corresponding SQL statement was skipped because processing was successful at some servers or an SQL error subject to skipping occurred. • The corresponding SQL statement was skipped because the `UPDATE` statement specifying the `SET` or `DELETE` clause containing an element number that is not in the repetition column was executed and a warning error occurred. • The system was unable to change at least one of the tables to check pending status; it ignored that processing and resumed normal processing.
		8	8	Abnormal termination
		12	12	Abnormal termination (an event disabling display of error messages occurred)
pdpfresh	KFPS00730-I	0	0	Normal termination
	—	—	8	Abnormal termination (such as invalid option or `rsh` error)

Command	Output message		Return code	Description
	Message ID	Code		
pdpgbfon	KFPL00738-I	0	0	Normal termination
		4	4	Normal termination (buffer miss occurred because there were not enough buffer sectors when data was read into the global buffer)
		8	8	Abnormal termination
		12	12	Abnormal termination (pdrorg terminated abnormally)
pdplgrgst	KFPY02016-I	0	0	Registration or deletion was successful.
		8	8	Registration or deletion failed.
pdplgset	KFPY01201-I	0	0	Setup was successful.
		8	8	Setup failed.
pdprgcopy	KFPS04645-I	0	0	Normal termination
		8	8	Abnormal termination
pdprgrenew	KFPS04646-I	0	0	Normal termination
		8	8	Abnormal termination (HiRDB status before updating)
		12	12	Abnormal termination (HiRDB terminated)
	KFPS05033-E	8	8	Abnormal termination (abnormal termination of system function)
pdrbal	KFPL33001-I	0	0	Normal termination
		4	4	Normal termination (rebalancing stopped temporarily)
		8	8	Abnormal termination
pdrbk	—	—	0	Normal termination
			1	Abnormal termination (such as invalid option or rsh error)

Command	Output message		Return code	Description
	Message ID	Code		
pdrdrefls	—	—	0	Normal termination
			4	Warning termination (there is no resource to be analyzed)
			8	Abnormal termination
pdreclaim	KFPL00739-I	0	0	Normal termination
		4	4	Normal termination (processing was cancelled because timeout occurred while waiting for settlement of UAP transaction)
		8	8	Abnormal termination
pdreginit	KFPX24601-I	0	0	Normal termination
		4	4	Normal termination (warning-level error occurred, but processing terminated normally)
		8	8	Abnormal termination
pdrels	—	—	0	Normal termination
			1	Warning termination At the updatable backup-hold (WAIT mode), the update buffer contents were applied to the RDAREA. Shutdown release processing has terminated normally.
			4	Warning termination (there is an error in at least one of the RDAREAs that were specified)
			8	Abnormal termination
			12	Abnormal termination (an event disabling display of error messages occurred)
pdrisechk	—	—	0	Normal termination
			8	Abnormal termination

Command	Output message		Return code	Description
	Message ID	Code		
`pdrisedbto`	—	—	0	Normal termination
			4	Some units terminated abnormally
			8	Abnormal termination
`pdriseset`	—	—	0	Normal termination
			8	Abnormal termination
`pdrorg`	`KFPL00719-I`	0	0	Normal termination
		4	4	Normal termination (some processes were skipped)
		8	8	Abnormal termination
`pdrpause`	—	—	0	Normal termination
			8	Abnormal termination (such as invalid option or `rsh` error)
`pdrplstart`	—	—	0	Normal termination
			8	Abnormal termination
`pdrplstop`	—	—	0	Normal termination
			4	Abnormal termination (some units were inactive (except ones that had not been started due to reduced activation), or some log information was not extracted by HiRDB Datareplicator)
			8	Abnormal termination
`pdrstr`	`KFPR00765-I`	0	0	Normal termination
		4	4	Warning termination (recovery processing terminated normally, but a warning level error that is unrelated to recovery processing occurred)
		8	8	Abnormal termination (an error occurred during recovery processing, but some RDAREAs were restored)
		12	12	Abnormal termination

Command	Output message		Return code	Description
	Message ID	Code		
`pdsetup`	—	—	0	Normal termination
	KFPS00011-I	8	1	Privilege error, file creation error, locking error, etc.
`pdstart`	KFPS01850-I KFPS05110-I	—	0	Normal termination
	—	—	4	Message was displayed due to timeout.
			8	Abnormal termination (invalid option)
`pdstbegin`	—	—	0	Normal termination
			8	Abnormal termination (such as invalid option or `rsh` error)
`pdstedit`	KFPK00300-I	0	0	Normal termination
		4	0	Warning termination
		8	1	Abnormal termination
`pdstend`	—	—	0	Normal termination
			8	Abnormal termination (such as invalid option or `rsh` error)
`pdstjswap`	—	—	0	Normal termination
			8	Abnormal termination (such as invalid option or `rsh` error)
`pdstjsync`	KFPS05840-I	—	0	Normal termination
	KFPS01841-E	—	8	Abnormal termination (invalid option or unavailable statistical information)
`pdstop`	—	—	0	Normal termination
			4	Abnormal termination (communication timeout during planned shutdown)
			8	Abnormal termination (invalid option)
`pdstscls`	—	—	0	Normal termination
			8	Abnormal termination (such as invalid option or `rsh` error)

Command	Output message		Return code	Description
	Message ID	Code		
pdstsinit	—	—	0	Normal termination
			8	Abnormal termination (such as invalid option or rsh error)
			12	Abnormal termination (when processing was retried at the standby system in a configuration without inheritance of IP addresses)
pdstsopen	—	—	0	Normal termination
			8	Abnormal termination (such as invalid option or rsh error)
pdstsrm	—	—	0	Normal termination
			8	Abnormal termination (such as invalid option or rsh error)
			12	Abnormal termination (when processing was retried at the standby system in a configuration without inheritance of IP addresses)
pdstsswap	—	—	0	Normal termination
			8	Abnormal termination (such as invalid option or rsh error)
	KFPS05210-I	—	0	Normal termination
pdsvhostname	—	—	0	Normal termination
			1	Normal termination (server is inactive)
			4	Abnormal termination (HiRDB is inactive)
			8	Abnormal termination (command was executed at a server machine where the system manager is not located, or the specified server was not found, etc.)
pdtrndec	—	—	0	Normal termination
			4	Warning termination (there is at least one unsettled transaction)
			8	Abnormal termination (invalid option)

Command	Output message		Return code	Description
	Message ID	Code		
`pdtrnqing`	—	—	`0`	Normal termination
			`4`	Warning termination (`pd_ha_transaction=queing` is missing in the system definition)
			`8`	Warning termination (invalid execution environment, operation error, etc.)
			`12`	Abnormal termination (other than the above)
`pdusrchk`	—	—	`0`	Normal termination
			`8`	Abnormal termination
`pdvrup`	`KFPX24404-I`	`0`	`0`	Normal termination
		`4`	`4`	Normal termination (an error occurred during execution of `pdvrup`, but upgrading was successful)
		`12`	`12`	Abnormal termination
`pdvwopt`	—	—	`0`	Normal termination
			`8`	Abnormal termination

Legend:

— : No message or code is displayed.

Index

Reader's Comment Form

We would appreciate your comments and suggestions on this manual. We will use these comments to improve our manuals. When you send a comment or suggestion, please include the manual name and manual number. You can send your comments by any of the following methods:

- Send email to your local Hitachi representative.
- Send email to the following address:
  WWW-mk@itg.hitachi.co.jp
- If you do not have access to email, please fill out the following information and submit this form to your Hitachi representative:

Manual name:	
Manual number:	
Your name:	
Company or organization:	
Street address:	
Comment:	

(For Hitachi use)