For UNIX Systems

Scalable Database Server

HiRDB Version 8

System Definition

3000-6-353(E)

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

ActiveX is a trademark of Microsoft Corp. in the U.S. and other countries.

■ Restrictions

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

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

Printed in Japan.

■ Edition history

Edition 1 (3000-6-353(E)): March, 2007

■ Copyright

Preface

This manual explains the system definition of HiRDB Version 8, a scalable database server program product.

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

This manual is based on the following manuals, which 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 is organized as follows:

Chapter *1. Overview*

Explains the organization of the HiRDB system definitions and the syntax rules for specifying the system definitions.

Chapter *2. System Common Definition*

Explains the system common definition.

Chapter *3. Unit Control Information Definition*

Explains the unit control information definition.

Chapter *4. Server Common Definition*

Explains the server common definition.

Chapter *5. Single Server Definition*

Explains the single server definition.

Chapter *6. Front-End Server Definition*

Explains the front-end server definition.

Chapter *7. Dictionary Server Definition*

Explains the dictionary server definition.

Chapter *8. Back-End Server Definition*

Explains the back-end server definition.

Chapter *9. UAP Environment Definition*

Explains the UAP environment definition.

Chapter *10. Foreign Server Information Definition (When a Foreign Server Is HiRDB)*

Explains the foreign server information definition when a foreign server is HiRDB. Create this definition for using the HiRDB External Data Access facility.

Chapter *11. Foreign Server Information Definition (When a Foreign Server Is XDM/ RD E2)*

Explains the foreign server information definition when a foreign server is XDM/ RD E2. Create this definition for using the HiRDB External Data Access facility.

Chapter *12. Foreign Server Information Definition (When a Foreign Server Is ORACLE)*

Explains the foreign server information definition when a foreign server is ORACLE. Create this definition for using the HiRDB External Data Access facility.

Chapter *13. Foreign Server Information Definition (When a Foreign Server Is DB2)*

Explains the foreign server information definition when a foreign server is DB2 (DB2 Universal Database for OS/390 Version 6). Create this definition for using the HiRDB External Data Access facility.

Chapter *14. Hub Optimization Information Definition*

Explains the Hub optimization information definition. Create this definition for using the HiRDB External Data Access facility.

Appendix *A. List of Operands*

Provides a list of operands and indicates the operands that can be changed during a restart.

Appendix *B. Operand Specification Values*

Explains the information to be specified in operands (arranged by types of operands).

Appendix *C. Examples of Definitions*

Provides examples of HiRDB system definitions.

Appendix *D. Formulas for Determining Operand Specification Values*

Provides and explains formulas that can be used to determine operand specification values.

Appendix *E. Determining the Number of Locked Resources*

Provides and explains formulas that can be used to determine the numbers of locked resources.

Appendix *F. Operands Checked by the pdconfchk Command*

Provides a list of operands that can be checked by executing the `pdconfchk` command.

Appendix *G. Operands Created by the pdgen Command*

Lists the operands created by the `pdgen` command.

Appendix *H. Operands Specifiable Under Each OS*

Lists the operands that can be specified for each operating system.

Appendix *I. List of Operands That Can Be Specified When Using the Standby-less System Switchover (Effects Distributed) Facility (Unit Control Information Definition)*

Lists the operands that can be specified in the unit control information definition for the standby-less system switchover (effects distributed) facility.

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 Operation Guide* (3000-6-354(E))
- *For UNIX Systems HiRDB Version 8 Command Reference* (3000-6-355(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.

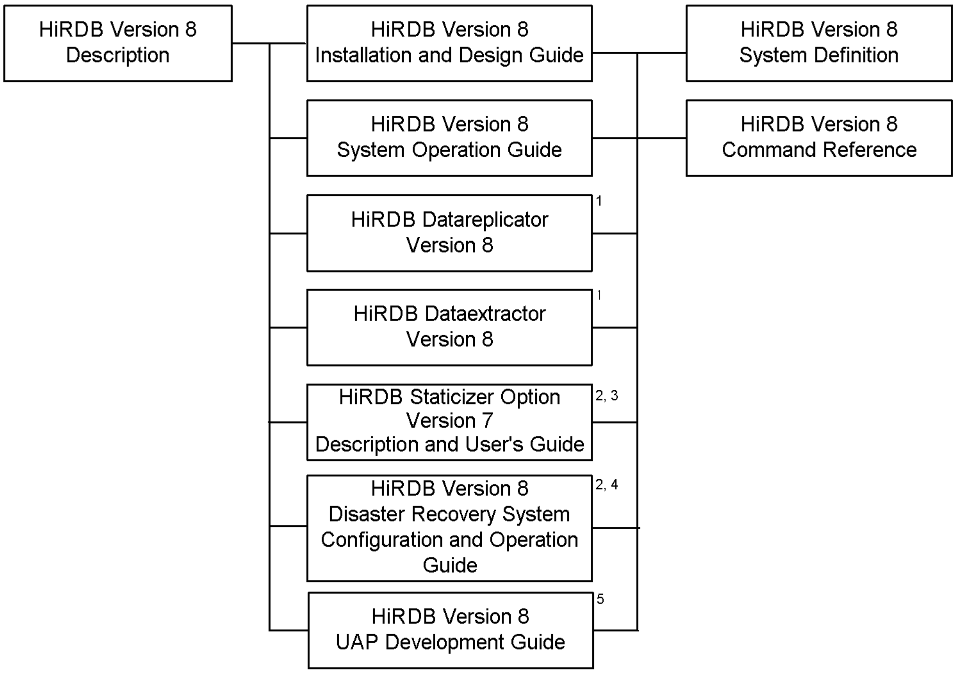
Others

- *HiRDB External Data Access Version 7 Description and User's Guide* (3000-6-284(E))
- *HI-UX/WE2 Hitachi CSMA/CD Network CD105 (TCP/IP)* (3000-3-200(E))

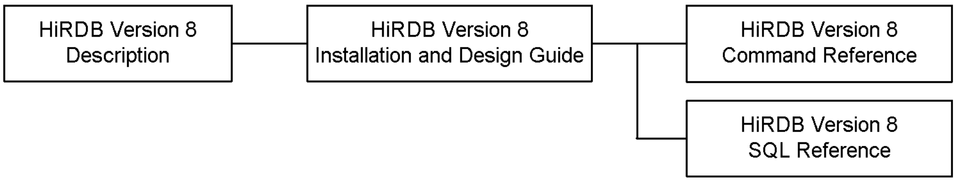
Organization of HiRDB manuals

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.

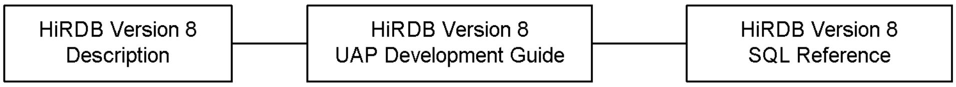
Manuals for system administrators:


HiRDB Version 8 Description
HiRDB Version 8 Installation and Design Guide
HiRDB Version 8 System Operation Guide
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
HiRDB Version 8 System Definition
HiRDB Version 8 Command Reference

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 8	HiRDB Datareplicator	
HiRDB Dataextractor Version 8	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	
$^{\text{DNCWARE}}$ ClusterPerfect (Linux Version)	ClusterPerfect	
Microsoft$_{\text{(R)}}$ Excel	Microsoft Excel or Excel	
Microsoft$_{\text{(R)}}$ Visual C++$_{\text{(R)}}$	Visual C++ or C++	
Oracle 8i	ORACLE	
Oracle 9i		
Oracle 10g		
Sun Java$^{\text{TM}}$ 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
Microsoft$_{(R)}$ Windows$_{(R)}$ XP Home Edition Operating System	Windows XP Home Edition	
Single server	SDS	

Name of product or other entity	Representation
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
COBOL	Common Business Oriented Language
CORBA(R)	Common ORB Architecture

Abbreviation	Full name or meaning
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
FTP	File Transfer Protocol
GUI	Graphical User Interface

Abbreviation	Full name or meaning
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
LUN	Logical Unit Number
LVM	Logical Volume Manager

Abbreviation	Full name or meaning
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
PR	Protected Retrieve
PU	Protected Update

Abbreviation	Full name or meaning
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
VOS1	Virtual-storage Operating System 1
VOS3	Virtual-storage Operating System 3

Abbreviation	Full name or meaning
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

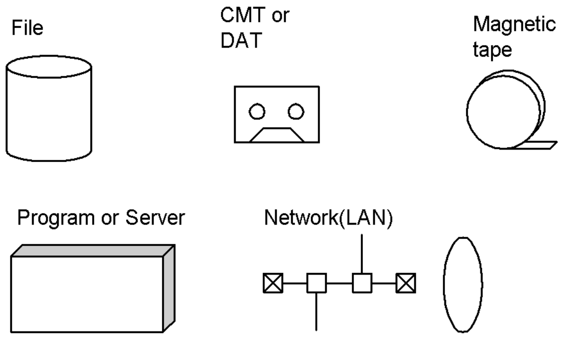
Log representations

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.
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:

Store**D**atabase [temp|perm] (*database-name* ...)

The following table lists and explains the symbols used in syntax definitions and explanations. When typing an actual command or definition, omit any syntax symbols listed here.

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.
perm	Underlined characters indicate the default value.
[ ]	Square brackets enclose an item or set of items whose specification is optional. Example: pdbuffer [-p] This example indicates that you can specify either pdbuffer or pdbuffer -p.
|	Only one of the options separated by a vertical bar can be specified at the same time. Example: pdlogadfg -d sys | spd This example indicates that you can specify either sys or spd for the -d option.
...	An ellipsis (...) indicates that the item or items enclosed in ( ) or [ ] immediately preceding the ellipsis may be specified as many times as necessary. Example: pdbuffer -r *RDAREA-name* [, *RDAREA-name*] ... This example indicates that you can specify *RDAREA-name* after -r as many times as you wish.
()	Parentheses indicate the range of items to which the vertical bar (|) or ellipsis (...) is applicable.
{ }	A single pair of curly brackets encloses multiple items, one of which you must specify. Example: pdbuffer [{-r *RDAREA-name* | -i *authorization-identifier*.*index-identifier* | -o}] This example indicates that you must specify any one of the three options enclosed by the curly brackets: -r *RDAREA-name*, -i *authorization-identifier*.*index-identifier*, or -o.

Example font or symbol	Convention
{{ }}	A double pair of curly brackets encloses multiple items, all of which you can specify multiple times as a single unit. Example: {{pdbuffer -a *option-name*}} This example indicates that you can specify these multiple times as follows: pdbuffer -a *option-name* pdbuffer -a *option-name*
_	An underlined value specifies a default. Example: [pd_rpc_trace = Y | N] If the operand is not specified, the default value (N) is used.
～	A swung dash precedes the attributes of a user-specified value.
<< >>	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 conventions

The following syntactical element symbols are used in this manual:

Syntactical element symbol	Meaning
<*alphabetics*>	The alphabetic characters (A-Z and a-z) and the underline (_)
<*alphabetics and special characters*>	The alphabetic characters (A-Z and a-z) and the special characters #, @, and \
<*alphanumerics*>	Alphabetics and the numeric digits (0-9)
<*alphanumerics and special characters*>	Alphabetics, special characters, and numeric digits
<*unsigned integer*>	Numeric value
<*unsigned decimal*>[1]	Numeric value (0-9), period (.), numeric value (0-9)
<*hexadecimal*>	Numeric digits and A-F (or a-f)
<*identifier*>[2]	Alphanumeric character string beginning with an alphabetic character
<*symbolic name*>	Alphanumeric character string beginning with an alphabetic character or a special character
<*character string*>	Any string of characters

Syntactical element symbol	Meaning
<pathname>[3]	Alphanumeric characters, /, and periods (.)
<host-name>	Character string composed of alphabetic characters (A-Z, a-z), numeric characters, periods (.), hyphens (-), or underscores (_)

Note: All alphabetic characters must be single-byte characters. The syntactical element symbols are case sensitive.

[1] If the numeric value preceding the period is 0, it can be omitted. Similarly, if the numeric value following the period is 0, both the period and the 0 can be omitted.

[2] A space may also be included in RDAREA names. However, when an RDAREA name includes a space, the entire name must be enclosed in quotation marks (").

[3] Path names depend on the OS being used. Do not use a backslash (\) in HiRDB file system area names.

Notations used in formulas

The following notations are used in computational expressions:

Symbol	Meaning
↑ ↑	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: 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

Contents

8. Back-End Server Definition 659

9. UAP Environment Definition 733

10. Foreign Server Information Definition (When a Foreign Server Is HiRDB)739

13. Foreign Server Information Definition (When a Foreign Server Is DB2) 793

14. Hub Optimization Information Definition 801

Appendixes 819

List of figures

List of tables

Chapter

1. Overview

This chapter explains the organization of the HiRDB system definitions and the syntax rules for specifying the system definitions.

This chapter contains the following sections:

1.1 For users of a HiRDB/Single Server

The HiRDB administrator creates HiRDB system definitions to set up a HiRDB execution environment. This section explains the following items:

- HiRDB system definition types
- HiRDB system definition creation procedure
- HiRDB system definition file structure
- Relationships among the definitions
- Procedure for modifying the HiRDB system definition (excluding the UAP environment definition)
- Procedure for adding or modifying the UAP environment definition

1.1.1 HiRDB system definition types

Table 1-1 lists the types of HiRDB system definitions. The HiRDB administrator must specify values for the operands of the definitions.

Table 1-1: Types of HiRDB system definitions (for a HiRDB/Single Server)

Definition type	Explanation
System common definition	Defines the structure of the HiRDB system and common information. Each HiRDB/Single Server requires a system common definition. For details about the system common definition operands, see Chapter *2. System Common Definition*.
Unit control information definition	Defines control information for a unit. Each unit requires a unit control information definition. For details about the unit control information definition operands, see Chapter *3. Unit Control Information Definition*.
Server common definition	Defines default values for single server definition operands. A server common definition is optional and is defined if needed. For details about the server common definition operands, see Chapter *4. Server Common Definition*.
Single server definition	Defines the execution environment of the single server. Each single server requires a single server definition. A utility special unit does not require a single server definition. For details about the single server definition operands, see Chapter *5. Single Server Definition*.
UAP environment definition	Defines the execution environment of a UAP. UAP environment definitions can be defined as needed. For details about the UAP environment definition operands, see Chapter *9. UAP Environment Definition*. You can create a maximum of 4,096 UAP environment definitions.

Definition type	Explanation
SQL reserved word definition	Defines SQL reserved words. Required when using the SQL reserved word deletion facility. For details about the SQL reserved word deletion facility, see *pd_delete_reserved_word_file operand.*

1.1.2 HiRDB system definition creation procedure

The HiRDB administrator uses one of the following tools to create HiRDB system definitions:

- simple setup tool
- System generator (`pdgen` command)
- OS text editor

Hint:

Normally, you will use the simple setup tool to create the HiRDB system definitions.

The created HiRDB system definitions should be stored in the files listed in Table 1-2. These files are referred to collectively as HiRDB system definition files.

Table 1-2: Storage files for HiRDB system definitions (HiRDB/Single Server)

Definition type	Storage file name
System common definition	`$PDDIR/conf/pdsys`
Unit control information definition	`$PDDIR/conf/pdutsys`
Server common definition[1]	`$PDDIR/conf/pdsvrc`
Single server definition	`$PDDIR/conf/`*server-name*[2]
UAP environment definition	`$PDDIR/conf/pduapenv/`*arbitrary-name*[3]
SQL reserved word definition	`$PDDIR/conf/pdrsvwd/`*arbitrary-name*[3]

[1] A server common definition is not created when the system generator is used to create the HiRDB system definitions. If a server common definition is needed, the HiRDB administrator must use the OS text editor to create it.

[2] This must be the server name specified in the `pdstart` operand of the system common definition.

If the simple setup tool is used to create the HiRDB system definitions, the file name is sds01.If the system generator is used to create the HiRDB system definitions, the file name is sds1.

3

- A file name must be an alphanumeric character string (a maximum of 8 characters) that begins with a letter.
- Assign to the users who will use the UAP environment definition or SQL reserved word definition the reading privilege (`r`) and execution privilege (`x`) for the directory in which each file is stored. Also assign the reading privilege (`r`) for the file.

Note:

Set and maintain the HiRDB system definition file permissions in such a manner that only the file owner (HiRDB administrator) has read and write permissions.

(1) simple setup tool

When the simple setup tool is employed to create HiRDB system definitions, the simple setup tool itself sets up the HiRDB environment. The HiRDB system definitions are created automatically on the basis of information specified with the simple setup tool and are stored in the files listed in Table 1-2. If necessary, the HiRDB administrator can use the simple setup tool to modify the definition contents (operand specification values).

Note that a UAP environment definition or SQL reserved word definition cannot be created using the simple setup tool.

(2) Using the system generator (pdgen command)

When the system generator is employed to create HiRDB system definitions, the system generator itself sets up the HiRDB environment. The HiRDB system definitions are created automatically on the basis of information specified with the system generator and are stored in the files listed in Table 1-2. If necessary, the HiRDB administrator can use the OS text editor to modify the definition contents (operand specification values). For details about the operands created by the `pdgen` command, see *G. Operands Created by the pdgen Command*.

Note that a UAP environment definition or SQL reserved word definition cannot be created using the system generator.

(3) Using an OS text editor

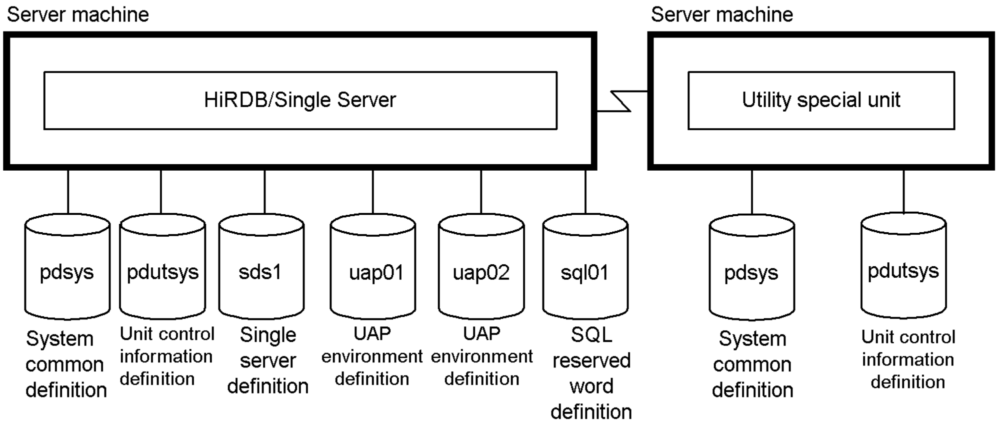
When an OS text editor is employed to create HiRDB system definitions, commands are used to set up the HiRDB environment. HiRDB system definition operands are specified as appropriate to the HiRDB execution environment.

Once the HiRDB system definitions have been created, use the `pdconfchk` command to check the integrity of the HiRDB system definition operands; see *F. Operands Checked by the pdconfchk Command* for the operands that are checked.

1.1.3 HiRDB system definition file structure

When the HiRDB system definitions are created with the simple setup tool or the system generator, HiRDB creates the HiRDB system definition files shown in Figure 1-1. Figure 1-1 should also be referenced when an OS text editor is used to create the HiRDB system definitions.

Figure 1-1: Structure of HiRDB system definition files (HiRDB/Single Server)



Note:
A utility special unit does not require a single server definition, UAP environment definition, or SQL reserved word definition.

1.1.4 Relationships among HiRDB system definitions

If the same operand is defined in different definitions, HiRDB determines the value to be used on the basis of the priority order of the definitions, as shown below:

1. Single server definition
2. Server common definition
3. Unit control information definition
4. System common definition

A standard value can be specified in a definition that has a lower priority, and then that value can be changed by specifying a different value in a definition that has a higher priority.

(1) Relationship to client environment definition

The values of the HiRDB system definition operands shown in Table 1-3 can be modified for a client in its client environment definition.

Table 1-3: Operands with specification values that can be modified in client environment definitions (HiRDB/Single Server)

HiRDB system definition operand	Client environment definition operand	Operand contents
pd_additional_optimize_level	PDADDITIONALOPTLVL	Specifies the SQL extension optimizing option.
pd_cwaittime_wrn_pnt	PDCWAITTIMEWRNPNT	Specifies the condition for outputting the SQL runtime warning information of the SQL runtime warning output facility as a percentage of the client's maximum wait time (value specified by the PDCWAITTIME operand in the client environment definition).
pd_delete_reserved_word_file	PDDELRSVWDFILE	Specifies the name of the SQL reserved word deletion file when using the SQL reserved word deletion facility.
pd_hash_table_size	PDHASHTBLSIZE	Specifies the hash table size when hash execution of a hash join or a subquery is used as the SQL extension optimizing option.
pd_hashjoin_hashing_mode	PDHJHASHINGMODE	Specifies the hashing method when hash join, subquery hash execution is specified as the SQL extension optimizing option.
pd_optimize_level	PDSQLOPTLVL	Specifies the SQL optimization option.
pd_space_level	PDSPACELVL	Specifies a space conversion level.
pd_uap_exerror_log_param_size	PDUAPEXERLOGPRMSZ	Specifies the maximum data size for the parameter information to be output to the error log file or SQL error report file.
pd_uap_exerror_log_use	PDUAPEXERLOGUSE	Specifies whether to use the facility for output of extended SQL error information.

HiRDB system definition operand	Client environment definition operand	Operand contents
pd_watch_pc_client_time	PDSWATCHTIME	Specifies the amount of time to wait for a request from a Windows-compatible HiRDB client.

To change the value for a client, specify the applicable operand in the client's environment definition. For details about client environment definition, see the *HiRDB Version 8 UAP Development Guide*.

1.1.5 Procedure for modifying the HiRDB system definitions (excluding the UAP environment definition)

(1) Modifying the HiRDB system definitions

This subsection describes how to modify the HiRDB system definitions. Note that `$PDDIR/conf` is the directory that stores the unit control information definition file. `$PDCONFPATH` is the directory that stores other HiRDB system definition files.

To modify the HiRDB system definitions:

1. Create subdirectories under `$PDDIR/conf` and `$PDCONFPATH`. In this example, a subdirectory called `work` is created.
2. Copy the unit control information definition file to `$PDDIR/conf/work`. Copy other HiRDB system definition files to `$PDCONFPATH/work`.
3. Modify the HiRDB system definitions that have been copied to `$PDDIR/conf/work` and `$PDCONFPATH/work`.
4. Use the `pdconfchk -d work` command to check the content of the HiRDB system definitions in `$PDDIR/conf/work` and `$PDCONFPATH/work`. If any error is found, correct the HiRDB system definition and re-execute the `pdconfchk` command.
5. Use the `pdstop` command to normally terminate HiRDB.
6. Use the `pdlogunld` command to unload the system log files that are waiting to be unloaded.
7. Replace the HiRDB system definition file by copying the HiRDB system definition file modified in Step 3 to `$PDDIR/conf` or `$PDCONFPATH`.
8. If you modify the specified values of the following operands, you must use the `pdloginit` command to initialize the system log file.
    - `pd_log_dual`
    - `pdstart`

9. Use the `pdstart` command to normally start the HiRDB.

(2) Using the system reconfiguration command to modify the HiRDB system definitions

If you use the system reconfiguration command (`pdchgconf` command), you can modify the HiRDB system definitions while HiRDB is running, and therefore you need not normally terminate HiRDB. Note however that HiRDB Advanced High Availability is required for using this command.

To modify the HiRDB system definition using the system reconfiguration command:

1. Create the `$PDDIR/conf/chgconf` directory.
2. Copy the HiRDB system definition file being used to the directory created in Step 1.
3. Modify the HiRDB system definitions in `$PDDIR/conf/chgconf`.
4. Use the `pdconfchk -d chgconf` command to check the HiRDB system definitions in `$PDDIR/conf/chgconf`. If any error is found, correct the HiRDB system definition and re-execute the `pdconfchk` command.
5. Use the `pdchgconf` command to replace the HiRDB system definition with the modified one.

   When you execute the `pdchgconf` command, the HiRDB system definition file being used (unmodified one) is saved to `$PDDIR/conf/backconf`. Then, the modified HiRDB system definition file in `$PDDIR/conf/chgconf` is copied to `$PDDIR/conf`.

Notes:

- If a transaction or utility continues to be active for 15 minutes or longer after the `pdchgconf` command is input, the command is terminated abnormally.
- Some limits apply when you use the system reconfiguration command to modify the HiRDB system definitions. For details on these limits, see the *HiRDB Version 8 System Operation Guide*.

(3) Notes

- HiRDB system definitions must not be modified or deleted while they are being used by an active HiRDB. If such a definition is modified or deleted, the operation of the HiRDB cannot be guaranteed.
- Once a planned, forced, or abnormal termination of HiRDB occurs, some of the HiRDB system definition operands cannot be modified; for details, see *A. List of Operands* or the operand explanations for the various definition statements.
- After you have modified the HiRDB system definitions, back up the files located

in $PDDIR/conf. To guard against errors in the disk containing the HiRDB directory, back up the files in the HiRDB directory (files in $PDDIR/conf). To restore the HiRDB directory, you need the backup of the files in $PDDIR/conf. Also back up $PDCONFPATH if it is located under the HiRDB directory.

(4) Linking a HiRDB Datareplicator

Before the operands listed below are added, modified, or deleted, HiRDB Datareplicator must be terminated; HiRDB Datareplicator can be restarted after the operand addition, modification, or deletion operation is completed:

- pd_log_dual
- pd_log_max_data_size
- pdlogadfg -d sys
- pdlogadpf -d sys

If these operands are added, modified, or deleted while HiRDB Datareplicator is running, extraction by HiRDB Datareplicator may fail under some circumstances.

1.1.6 Adding or modifying the UAP environment definition

To add or modify the UAP environment definition:

1. Make sure that the UAP that uses the UAP environment definition to be modified is not active. If it is active, wait until it is terminated. If a UAP environment definition is added or modified while the UAP is active, the unmodified UAP environment definition is applied to the active UAP. However, depending on the timing, the modified UAP environment definition may be applied to the UAP.
2. Add or modify the UAP environment definition.
3. Use the modified UAP environment definition to execute the UAP.

1.2 For users of a HiRDB/Parallel Server

The HiRDB administrator creates HiRDB system definitions to set up a HiRDB execution environment. This section explains the following items:

- HiRDB system definition types
- HiRDB system definition creation procedure
- HiRDB system definition file structure
- Relationships among the definitions
- Procedure for modifying the HiRDB system definition (excluding the UAP environment definition)
- Procedure for creating a HiRDB system definition when using the standby-less system switchover (effects distributed) facility
- Sharing of HiRDB system definition files

1.2.1 HiRDB system definition types

Table 1-4 lists the types of HiRDB system definitions. The HiRDB administrator must specify values for the operands of the definitions.

Table 1-4: Types of HiRDB system definitions (for a HiRDB/Parallel Server)

Definition type	Explanation
System common definition	Defines the structure of the HiRDB system and common information. Each unit requires a system common definition. *The contents of the individual system common definitions must be identical for all units.* For details about the system common definition operands, see Chapter *2. System Common Definition*.
Unit control information definition	Defines control information for a unit. Each unit requires a unit control information definition. For details about the unit control information definition operands, see Chapter *3. Unit Control Information Definition*.

Definition type		Explanation
Server definitions	Server common definition	Defines default values for server definition operands. A server common definition is optional and is defined if needed. For details about the server common definition operands, see Chapter *4. Server Common Definition*.
	Front-end server definition	Defines the execution environment of a front-end server. Each front-end server requires a front-end server definition. For details about the front-end server definition operands, see Chapter *6. Front-End Server Definition*.
	Dictionary server definition	Defines the execution environment of a dictionary server. Each dictionary server requires a dictionary server definition. For details about the dictionary server definition operands, see Chapter *7. Dictionary Server Definition*.
	Back-end server definition	Defines the execution environment of a back-end server. Each back-end server requires a back-end server definition. For details about the back-end server definition operands, see Chapter *8. Back-End Server Definition*.
UAP environment definition		Defines the execution environment of a UAP. UAP environment definitions can be defined as needed. Create the UAP environment definition in the unit in which a front-end server is located. If there are multiple front-end servers, create the definition for the front-end server to which the UAP environment definition is to be applied. For details about the UAP environment definition operands, see Chapter *9. UAP Environment Definition*. You can create a maximum of 4,096 UAP environment definitions.
SQL reserved word definition		Required when using the SQL reserved word deletion facility. Defines SQL reserved words. Create the SQL reserved word definition in the unit in which a front-end server is located. If there are multiple front-end servers, you must create the same SQL reserved word definition for all units in which the front-end servers are located. For details about the SQL reserved word deletion facility, see *pd_delete_reserved_word_file operand*.
Foreign server information definition		Defines a connection environment to a foreign server. Each foreign server to be connected requires a connection environment. Create the connection environment definition in the unit in which the back-end server for connecting to the foreign server is located. A connection environment definition is required for using the HiRDB External Data Access facility. For details about the foreign server information definition operands, see the following chapters: • *10. Foreign Server Information Definition (When a Foreign Server Is HiRDB)* • *11. Foreign Server Information Definition (When a Foreign Server Is XDM/RD E2)* • *12. Foreign Server Information Definition (When a Foreign Server Is ORACLE)* • *13. Foreign Server Information Definition (When a Foreign Server Is DB2)*

Definition type	Explanation
Hub optimization information definition	Defines the optimization information for the HiRDB External Data Access facility. Create the optimization information definition in the unit in which the front-end server is located. If there are multiple front-end servers, all units must have the same Hub optimization information definition. A Hub optimization information definition is required for using the HiRDB External Data Access facility. For details about the Hub optimization information definition operands, see Chapter *14. Hub Optimization Information Definition*.

1.2.2 HiRDB system definition creation procedure

The HiRDB administrator uses one of the following tools to create HiRDB system definitions:

- simple setup tool
- System generator (`pdgen` command)
- OS text editor

Hint:

Normally, you will use the simple setup tool to create the HiRDB system definitions.

The created HiRDB system definitions should be stored in the files listed in Table 1-5. These files are referred to collectively as *HiRDB system definition files*.

Table 1-5: Storage files for HiRDB system definitions (HiRDB/Parallel Server)

Definition type	Storage file name
System common definition	`$PDDIR/conf/pdsys`
Unit control information definition	`$PDDIR/conf/pdutsys`
Server common definition	`$PDDIR/conf/pdsvrc`
Front-end server definition	`$PDDIR/conf/`*server-name*[1, 2]
Dictionary server definition	`$PDDIR/conf/`*server-name*[1, 3]
Back-end server definition	`$PDDIR/conf/`*server-name*[1, 4]
UAP environment definition	`$PDDIR/conf/pduapenv/`*arbitrary-name*[5]
SQL reserved word definition	`$PDDIR/conf/pdrsvwd/`*arbitrary-name*[5]
foreign server information definition	`$PDDIR/conf/`*foreign-server-name*

Definition type	Storage file name
Hub optimization information definition	`$PDDIR/conf/`*arbitrary-name*

[1] A server name specified by the `-s` option of the `pdstart` operand of the system common definition is specified here.

[2] When the system generator is used to create the HiRDB system definitions, the file names are f001-fxxx, where xxx is a serial number in the range 1-999.

[3] When the system generator is used to create the HiRDB system definitions, the file name is dic.

[4] When the system generator is used to create the HiRDB system definitions, the file names are b001-bxxx, where xxx is a serial number in the range 1-999.

[5]

- A file name must be an alphanumeric character string (a maximum of 8 characters) that begins with a letter.
- Assign to the users who will use the UAP environment definition or SQL reserved word definition the reading privilege (`r`) and execution privilege (`x`) for the directory in which each file is stored. Also assign the reading privilege (`r`) for the file.

Note:

Set and maintain the HiRDB system definition file permissions in such a manner that only the file owner (HiRDB administrator) has read and write permissions.

(1) Using the simple setup tool

When the simple setup tool is employed to create HiRDB system definitions, the simple setup tool itself sets up the HiRDB environment. The HiRDB system definitions are created automatically on the basis of information specified with the simple setup tool and are stored in the files listed in Table 1-5. If necessary, the HiRDB administrator can use the simple setup tool to modify the definition contents (operand specification values).

Note that the following definitions cannot be created using the simple setup tool:

- UAP environment definition
- SQL reserved word definition
- Foreign server information definition

- Hub optimization information definition

(2) Using the system generator (pdgen command)

When the system generator is employed to create HiRDB system definitions, the system generator itself sets up the HiRDB environment. The HiRDB system definitions are created automatically on the basis of information specified with the system generator and are stored in the files listed in Table 1-5. If necessary, the HiRDB administrator can use the OS text editor to modify the definition contents (operand specification values). For details about the operands created by the `pdgen` command, see *G. Operands Created by the pdgen Command*.

Note that the following definitions cannot be created using the system generator:

- UAP environment definition
- SQL reserved word definition
- Foreign server information definition
- Hub optimization information definition

(3) Using an OS text editor

When an OS text editor is employed to create HiRDB system definitions, commands are used to set up the HiRDB environment. HiRDB system definition operands are specified as appropriate to the HiRDB execution environment.

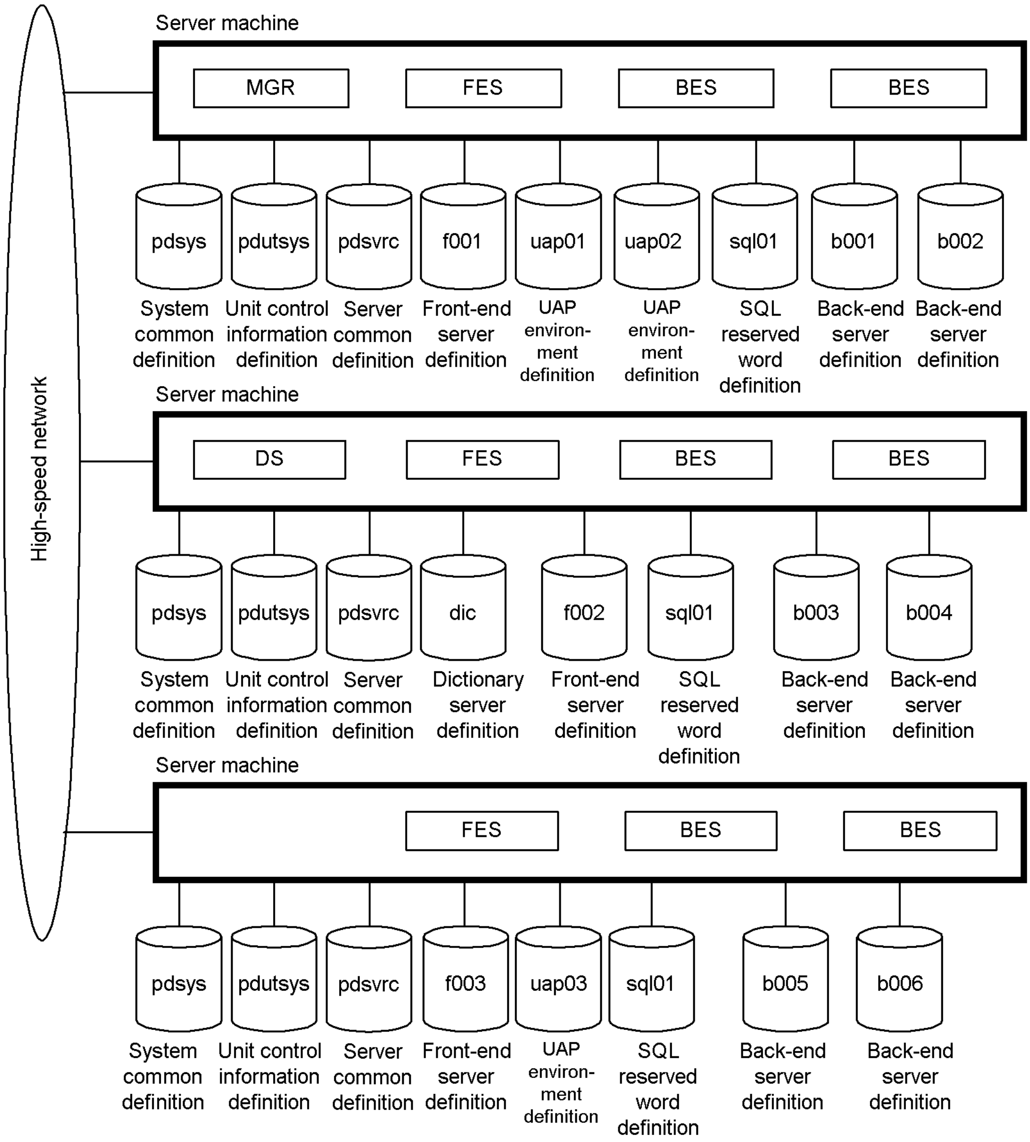
Once the HiRDB system definitions have been created, use the `pdconfchk` command to check the integrity of the HiRDB system definition operands; see *F. Operands Checked by the pdconfchk Command* for the operands that are checked.

1.2.3 HiRDB system definition file structure

When the HiRDB system definitions are created with the simple setup tool or the system generator, HiRDB creates the HiRDB system definition files shown in Figure 1-2. Figure 1-2 should also be referenced when an OS text editor is used to create the HiRDB system definitions.

Figure 1-2: Structure of HiRDB system definition files (HiRDB/Parallel

Server)


High-speed network
Server machine
MGR
FES
BES
BES
pdsys
pdutsys
pdsvrc
f001
uap01
uap02
sql01
b001
b002
System common definition
Unit control information definition
Server common definition
Front-end server definition
UAP environ-ment definition
UAP environ-ment definition
SQL reserved word definition
Back-end server definition
Back-end server definition
Server machine
DS
FES
BES
BES
pdsys
pdutsys
pdsvrc
dic
f002
sql01
b003
b004
System common definition
Unit control information definition
Server common definition
Dictionary server definition
Front-end server definition
SQL reserved word definition
Back-end server definition
Back-end server definition
Server machine
FES
BES
BES
pdsys
pdutsys
pdsvrc
f003
uap03
sql01
b005
b006
System common definition
Unit control information definition
Server common definition
Front-end server definition
UAP environ-ment definition
SQL reserved word definition
Back-end server definition
Back-end server definition

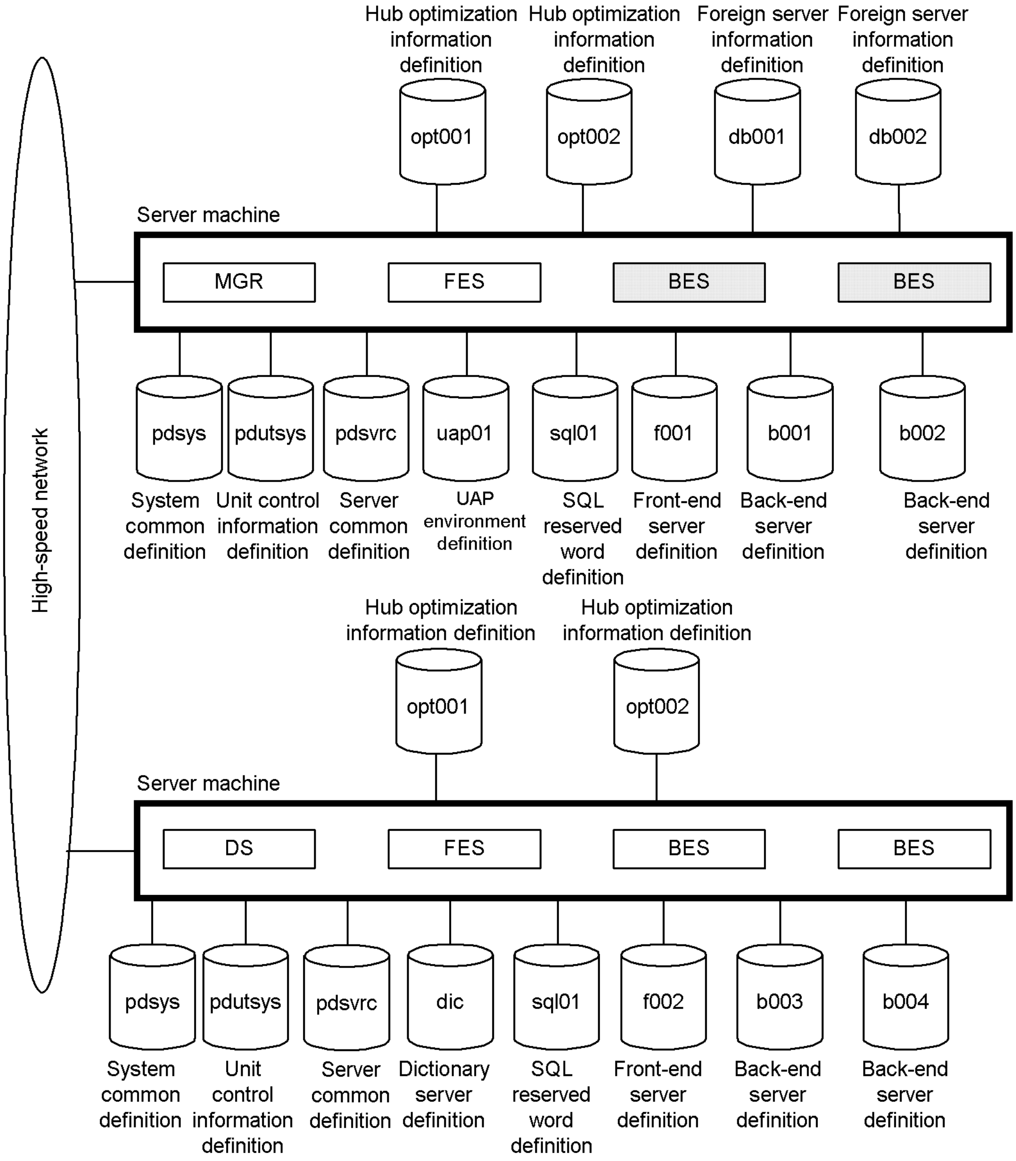
Note 1: The system common definition must be the same for all server machines.
Note 2: The SQL reserved word definition must be the same for all server machines.

■ When the HiRDB External Data Access facility is used

Figure 1-3 shows a configuration example of the HiRDB system definition file when the HiRDB External Data Access facility is used.

Figure 1-3: Structure of HiRDB system definition file (when the HiRDB

External Data Access facility is used)


Hub optimization information definition
Hub optimization information definition
Foreign server information definition
Foreign server information definition
opt001
opt002
db001
db002
Server machine
MGR
FES
BES
BES
pdsys
pdutsys
pdsvrc
uap01
sql01
f001
b001
b002
System common definition
Unit control information definition
Server common definition
UAP environment definition
SQL reserved word definition
Front-end server definition
Back-end server definition
Back-end server definition
High-speed network
Hub optimization information definition
Hub optimization information definition
opt001
opt002
Server machine
DS
FES
BES
BES
pdsys
pdutsys
pdsvrc
dic
sql01
f002
b003
b004
System common definition
Unit control information definition
Server common definition
Dictionary server definition
SQL reserved word definition
Front-end server definition
Back-end server definition
Back-end server definition

Note 1: The system common definition must be the same for all server machines.
Note 2: Shaded BESs are the back-end servers for connecting foreign servers.
Note 3: The SQL reserved word definition must be the same for all server machines.

1.2.4 Relationships among HiRDB system definitions

If the same operand is defined in different definitions, HiRDB determines the value to be used on the basis of the priority order of the definitions, as shown below:

1. Front-end server definition, dictionary server definition, or back-end server definition
2. Server common definition
3. Unit control information definition
4. System common definition

A standard value can be specified in a definition of a lower priority, and then that value can be changed by specifying a different value in a definition of a higher priority. For example, a standard value can be specified in the system common definition and then this value can be changed by specifying values in the server definitions of individual servers.

(1) Relationship to client environment definition

The values of the HiRDB system definition operands shown in Table 1-6 can be modified for a client in its client environment definition.

Table 1-6: Operands with specification values that can be modified in client environment definitions (HiRDB/Parallel Server)

HiRDB system definition operand	Client environment definition operand	Operand contents
pd_additional_optimize_level	PDADDITIONALOPTLVL	Specifies the SQL extension optimizing option.
pd_cwaittime_wrn_pnt	PDCWAITTIMEWRNPNT	Specifies the condition for outputting the SQL runtime warning information of the SQL runtime warning output facility as a percentage of the client's maximum wait time (value specified by the PDCWAITTIME operand in the client environment definition).
pd_delete_reserved_word_file	PDDELRSVWDFILE	Specifies the name of the SQL reserved word deletion file when using the SQL reserved word deletion facility.
pd_ha_transaction	PDHATRNQUEUING	Specifies whether to use the transaction queuing facility.

HiRDB system definition operand	Client environment definition operand	Operand contents
pd_hash_table_size	PDHASHTBLSIZE	Specifies the hash table size when hash execution of a hash join or a subquery is used as the SQL extension optimizing option.
pd_hashjoin_hashing_mode	PDHJHASHINGMODE	Specifies the hashing method when hash join, subquery hash execution is specified as the SQL extension optimizing option.
pd_optimize_level	PDSQLOPTLVL	Specifies the SQL optimization option.
pd_space_level	PDSPACELVL	Specifies a space conversion level.
pd_uap_exerror_log_param_size	PDUAPEXERLOGPRMSZ	Specifies the maximum data size for the parameter information to be output to the error log file or SQL error report file.
pd_uap_exerror_log_use	PDUAPEXERLOGUSE	Specifies whether to use the facility for output of extended SQL error information.
pd_watch_pc_client_time	PDSWATCHTIME	Specifies the amount of time to wait for a request from a Windows-compatible HiRDB client.

To change the value for a client, the applicable operand is specified in the client's environment definition. For details about client environment definition, see the *HiRDB Version 8 UAP Development Guide*.

1.2.5 Procedure for modifying the HiRDB system definitions (excluding UAP environment definition)

(1) Modifying the HiRDB system definitions

This subsection describes how to modify the HiRDB system definitions. Note that `$PDDIR/conf` is the directory that stores the unit control information definition file. `$PDCONFPATH` is the directory that stores other HiRDB system definition files.

To modify the HiRDB system definitions:

1. Create subdirectories under `$PDDIR/conf` and `$PDCONFPATH`. In this

example, a subdirectory called `work` is created.

2. Copy the unit control information definition file to `$PDDIR/conf/work`. Copy other HiRDB system definition files to `$PDCONFPATH/work`.
3. Modify the HiRDB system definitions that have been copied to `$PDDIR/conf/work` and `$PDCONFPATH/work`.
4. Use the `pdconfchk -d work` command to check the content of the HiRDB system definitions in `$PDDIR/conf/work` and `$PDCONFPATH/work`. If any error is found, correct the HiRDB system definition and re-execute the `pdconfchk` command.
5. Use the `pdstop` command to normally terminate HiRDB.
6. Use the `pdlogunld` command to unload the system log files that are waiting to be unloaded.
7. Replace the HiRDB system definition file by copying the HiRDB system definition file modified in Step 3 to `$PDDIR/conf` or `$PDCONFPATH`.
8. If you modify the specified values of the following operands, you must use the `pdloginit` command to initialize the system log file.
    - `pd_log_dual`
    - `pdstart`
9. Use the `pdstart` command to normally start HiRDB.

(2) Using the system reconfiguration command to modify the HiRDB system definitions

If you use the system reconfiguration command (`pdchgconf` command), you can modify the HiRDB system definitions while HiRDB is running, and therefore you need not normally terminate HiRDB. Note however that HiRDB Advanced High Availability is required for using this command.

To modify the HiRDB system definition using the system reconfiguration command:

1. Create the `$PDDIR/conf/chgconf` directory.
2. Copy the HiRDB system definition file being used to the directory created in Step 1.
3. Modify the HiRDB system definitions in `$PDDIR/conf/chgconf`.
4. Use the `pdconfchk -d chgconf` command to check the HiRDB system definitions in `$PDDIR/conf/chgconf`. If any error is found, correct the HiRDB system definition and re-execute the `pdconfchk` command.
5. Use the `pdchgconf` command to replace the HiRDB system definition with

the modified one.

When you execute the `pdchgconf` command, the HiRDB system definition file being used (unmodified one) is saved to `$PDDIR/conf/backconf`. Then, the modified HiRDB system definition file in `$PDDIR/conf/chgconf` is copied to `$PDDIR/conf`.

Notes:

- If a transaction or utility continues to be active for 15 minutes or longer after the `pdchgconf` command is input, the command is aborted.
- Some limits apply when you use the system reconfiguration command to modify the HiRDB system definitions. For details on these limits, see the *HiRDB Version 8 System Operation Guide*.

(3) Notes

- When a modification is made to a system common definition, the identical modification must be made to the system common definitions at all server machines.
- If some of the units are abnormally terminated during normal or planned termination of HiRDB, do not modify the HiRDB system definition before restarting HiRDB. If it is modified, HiRDB will not start. Even if HiRDB starts, it cannot run normally after the startup.
- For a HiRDB/Parallel Server, create subdirectories under `$PDDIR/conf` and `$PDCONFPATH` for each unit, and check the content of the HiRDB system definitions.
- HiRDB system definitions must not be modified or deleted while they are being used by an active HiRDB. If such a definition is modified or deleted, the operation of the HiRDB cannot be guaranteed.
- Once a planned, forced, or abnormal termination of HiRDB occurs, some of the HiRDB system definition operands cannot be modified; for details, see *A. List of Operands* or the operand explanations for the various definition statements.
- After you have modified the HiRDB system definitions, back up the files located in `$PDDIR/conf`. To guard against errors in the disk containing the HiRDB directory, back up the files in the HiRDB directory (files in `$PDDIR/conf`). To restore the HiRDB directory, you need the backup of the files in `$PDDIR/conf`. Also back up `$PDCONFPATH` if it is located under the HiRDB directory.
- Note the following when using the standby-less system switchover (1:1) facility: Before modifying the HiRDB system definitions of a normal BES unit, first use the `pdstop -u` command to normally terminate the normal BES unit and the alternate BES unit. After modifying the HiRDB system definitions, copy both the unit control information definition file and back-end server definition file of the

normal BES unit to the alternate BES unit. For details, see the section *Creating HiRDB system definitions (system switchover)* in the *HiRDB Version 8 System Operation Guide*.

(4) When linking a HiRDB Datareplicator

Before the operands listed below are added, modified, or deleted, HiRDB Datareplicator must be terminated; HiRDB Datareplicator can be restarted after the operand addition, modification, or deletion operation is completed:

- `pd_log_dual`
- `pd_log_max_data_size`
- `pdlogadfg -d sys`
- `pdlogadpf -d sys`

If these operands are added, modified, or deleted while HiRDB Datareplicator is running, extraction by HiRDB Datareplicator may fail under some circumstances.

1.2.6 Adding or modifying the UAP environment definition

To add or modify the UAP environment definition:

1. Make sure that the UAP that uses the UAP environment definition to be modified is not active. If it is active, wait until it is terminated. If a UAP environment definition is added or modified while the UAP is active, the unmodified UAP environment definition is applied to the active UAP. However, depending on the timing, the modified UAP environment definition may be applied to the UAP.
2. Add or modify the UAP environment definition.
3. Use the modified UAP environment definition to execute the UAP.

1.2.7 Creating a HiRDB system definition when using the standby-less system switchover (effects distributed) facility

This section explains how to create a HiRDB system definition when using the standby-less system switchover (effects distributed) facility and provides related notes.

Key point

The operating environment of the back-end server for the unit in which the host BES is running must match the operating environment of the back-end server for the unit in which the guest BES is running.

(1) System common definition

Use the same system common definition for all units. Copy the system common definition that has been created.

Note:

The operand to be specified as the default value of the back-end server definition should be specified in the system common definition. An example of such an operand is the `pd_sql_object_cache_size` operand. You cannot specify this operand in the server common definition only.

(2) Unit control information definition

Some restrictions apply to the operands that can be specified in the unit control information definition of the unit to which the standby-less system switchover (effects distributed) facility is to be applied. For details about the operands that can be specified in the unit control information definition, see *I. List of Operands That Can Be Specified When Using the Standby-less System Switchover (Effects Distributed) Facility (Unit Control Information Definition)*.

If you must specify an operand that cannot be specified in the unit control information definition, specify it in the system common definition or back-end server definition.

Notes:

If an operand satisfies either of the following conditions, it must be specified in the same way in all of the unit control information definitions of the units that comprise an HA group.

- Operand that can be specified in both the unit control information definition and back-end server definition
- Operand in the unit control information definition that affects the operation of the back-end server

(3) Server common definition

Units that comprise an HA group use the same server common definition. Copy the server common definition that has been created.

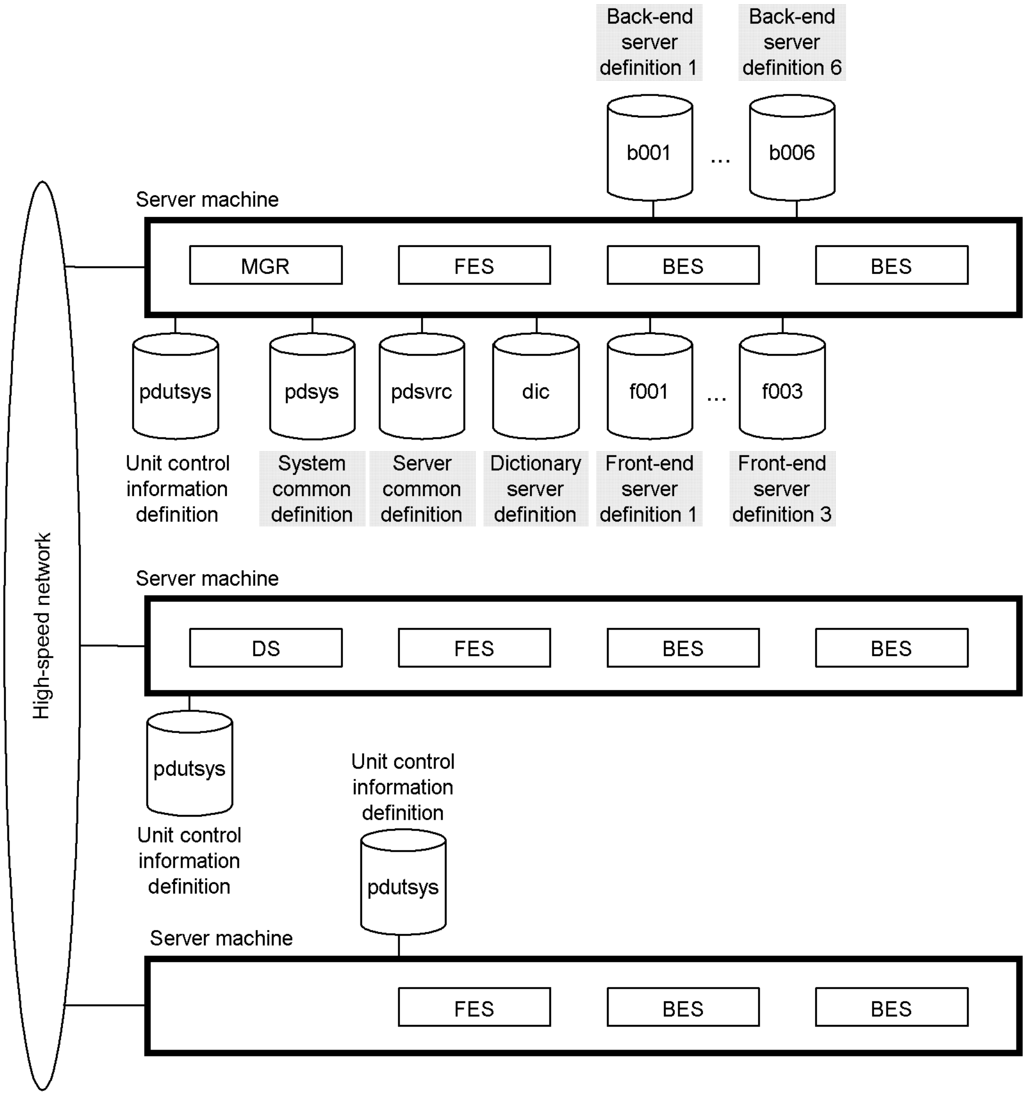
(4) Back-end server definition

Define the back-end server definition of the host BES in each of the units that comprise an HA group. Copy the back-end server definition that has been created.

1.2.8 Sharing of HiRDB system definition files

If the file sharing facility (NFS) of the Hitachi CSMA/CD network CD105 (TCP/IP) is used, it is possible to manage all HiRDB system definition files (except for unit control information definition files) from a single server machine. This is called *sharing of HiRDB system definition files*. Figure 1-4 shows sharing of HiRDB system definition files.

Figure 1-4: Sharing of HiRDB system definition files


Back-end server definition 1
Back-end server definition 6
b001
...
b006
Server machine
MGR
FES
BES
BES
pdutsys
pdsys
pdsvrc
dic
f001
...
f003
Unit control information definition
System common definition
Server common definition
Dictionary server definition
Front-end server definition 1
Front-end server definition 3
High-speed network
Server machine
DS
FES
BES
BES
pdutsys
Unit control information definition
Unit control information definition
pdutsys
Server machine
FES
BES
BES

Explanation

The shaded files are shared files.

(1) File sharing benefits

Not shared	Shared
HiRDB system definitions must be managed separately for each server machine.	HiRDB system definitions can be managed from a single server machine (except for the unit control information definitions).
When a system common definition must be modified, the same modification must be made for all server machines. For example, if there are four server machines, there are four system common definitions. Because the contents of the system common definitions must be identical, all four must be modified.	• Because there is only one system common definition, a modification need be made only once. • Because the number of modifications is reduced, there is a lower likelihood of errors. • If HiRDB cannot reference the shared directory, HiRDB might terminate abnormally.

(2) File sharing procedure

Procedure

1. Create a directory to store the HiRDB system definition files to be shared. This directory must be created at the server machine from which the HiRDB system definition files will be centrally managed. This directory is called the *shared directory*. Although any of the server machines can be used for central management, it is recommended that the server machine where the system manager is defined be used.
2. Create the HiRDB system definition file, excluding the unit control information definition file, under the shared directory (`$PDCONFPATH`). Create the UAP environment definition file under `$PDCONFPATH/pduapenv`.
3. Create a unit control information definition file under `$PDDIR/conf` of each server machine. During this process, specify the name of the common directory in the `PDCONFPATH` operand.

1.3 Coding format of HiRDB system definitions

This section explains the syntax rules for coding the HiRDB system definition operands.

(1) Operand specification format

HiRDB system definition operands are specified in the following three formats:

set format

`set` is used to set a value in an operand; for example,

```
set pd_max_users=15
```

Command format

Options and command arguments are set in an operand; for example,

```
pdlogadfg -d sys -g loggrp01 ONL
```

`pdlogadfg`

Command name

`-d sys -g loggrp01`

Options (an option is a character string that begins with a hyphen)

Flag arguments are not specified in Format 1; flag arguments are specified in Format 2:

- Format 1: *option-flag*
- Format 2: *option-flag flag-argument*

  option-flag: An alphabetic character that follows the hyphen.

  Values are case sensitive.

  flag-argument: Operation target for the option flag.

`ONL`

A command argument (argument that begins with anything other than a hyphen).

putenv format

`putenv` is used to set in an operand an environment variable and value for the environment variable; for example,

```
putenv SHMMAX 16
```

(2) Comments

A comment can be entered for any operand. A comment must begin with the number sign (#). When the number sign is encountered, the remainder of the line is assumed to be a comment. When the number sign is entered at the beginning of a line, the entire line is handled as a comment line.

Example

```
set pd_max_users=15          # maximum number of concurrent connections
pdlogadfg -d sys -g loggrp01 ONL
# Defines a file group for system log files
```

(3) Line continuation

The maximum number of characters per line of definition is 80. If a definition requires more than 80 characters, additional lines can be used by specifying the backslash (\) as a continuation symbol before the end of each line that is to be continued.

Example

```
pdbuffer -a buffer ABC -n 160 -r rdareaA,rdareaB,..... \
rdareaZ
```

A line in which a comment is specified cannot be continued. When the number sign (#) is encountered, the entire remainder of the line is handled as a comment line, so even if the backslash is specified subsequently on the line, it will be regarded as part of the comment rather than as the line continuation symbol.

(4) Duplicate operand specification

When multiple operands are specified in a single definition statement (single file), HiRDB uses the following rules for processing these operands:

- When the operands are in the `set` or `putenv` format, the value of the operand that was specified last takes precedence.
- When the operands are in the command format, priority depends on the operands. Refer to the explanations of the individual operands.

1.4 Operands whose default value depends on the version and operands that are no longer needed

(1) Operands whose default value depends on the version

Table 1-7 lists and describes the operands whose default value depends on the version.

If you do not want to change operand default values during upgrading, specify `v6compatible` or `v7compatible` in the `pd_sysdef_default_option` operand.

Table 1-7: Operands whose default value depends on the version

Operand name	**Recommendable** **(default value in 08-00 and newer versions)**	**v7compatible** **(default value in 07-00)**	**v6compatible** **(default value in versions older than 07-00)**
`pd_additional_optimize_level`	`"COST_BASE_2"`	`"COST_BASE_2"`	`"NONE"`
`pd_assurance_index_no`	`500` (automatic addition is not made during normal startup of HiRDB)	• For HiRDB/Single Server: Number of indexes for data dictionary table (124) + 50 (added automatically during normal startup of HiRDB) • For HiRDB/Parallel Server: `50` (added automatically during normal startup of HiRDB)	• For HiRDB/ Single Server: Number of indexes for data dictionary table (124) + 50 (added automatically during normal startup of HiRDB) • For HiRDB/ Parallel Server: `50` (added automatically during normal startup of HiRDB)
`pd_assurance_table_no`	`500` (automatic addition is not made during normal startup of HiRDB)	`100` (automatic addition is made during normal startup of HiRDB)	`0` (automatic addition is not made during normal startup of HiRDB)
`pd_aud_async_buff_count`	MAX(1, number of HiRDB servers in unit x 10)	`3`	`3`
`pd_aud_async_buff_size`	`401408`	`4096`	`4096`

Operand name	Recommendable (default value in 08-00 and newer versions)	v7compatible (default value in 07-00)	v6compatible (default value in versions older than 07-00)
pd_bes_shmpool_size	Automatic computation by HiRDB	Automatic computation by HiRDB	1024
pd_cancel_down_msgchange	Y	N	N
pd_dbbuff_attribute	free	fixed (free in the 64-bit mode)	fixed (free in the 64-bit mode)
pd_dbbuff_lru_option	MIX	SEPARATE	SEPARATE
pd_dic_shmpool_size	Automatic computation by HiRDB	Automatic computation by HiRDB	1024
pd_lck_hash_entry	0 (Automatic computation by HiRDB)	0 (Automatic computation by HiRDB)	11261
pd_lck_pool_size	16000 (32000 in the 64-bit mode)	16000 (32000 in the 64-bit mode)	1024
pd_lck_release_detect	pipe	interval	interval
pd_lck_wait_timeout	MAX (180, value of pd_watch_time)	MAX (180, value of pd_watch_time)	180
pd_log_max_data_size	400000	32000	32000
pd_log_sdinterval	• *system-log-output-volume*: 5000 • interval: 60	• *system-log-output-volume*: 1000 • interval: 60	• *system-log-output-volume*: 1000 • interval: 60
pd_log_write_buff_count	10	3	3
pd_max_access_tables	64	64	16
pd_max_commit_write_reclaim_no	10	0	0
pd_max_server_process	Automatic computation by HiRDB	Automatic computation by HiRDB	100
pd_optimize_level	Refer to the pd_optimize_level operand.	Refer to the pd_optimize_level operand.	"SELECT_APSL"
pd_pageaccess_mode	SNAPSHOT	SNAPSHOT	NORMAL

Operand name	Recommendable (default value in 08-00 and newer versions)	v7compatible (default value in 07-00)	v6compatible (default value in versions older than 07-00)
pd_process_terminator	fixed	fixed	nonresident
pd_process_terminator_max	Max(3, ↑(Values of pd_max_users operand + Values of pd_max_reflect_process_count operand) ÷ 100↑)	↑(Values of pd_max_users operand + Values of pd_max_reflect_process_count operand) ÷ 100↑	↑(Values of pd_max_users operand + Values of pd_max_reflect_process_count operand) ÷ 100↑
pd_sds_shmpool_size	Automatic computation by HiRDB	Automatic computation by HiRDB	1024
pd_spool_cleanup_interval_level	• *number-of-days*: 7 • *deletion-type*: all	• *number-of-days*: 7 • *deletion-type*: dump	• *number-of-days*: 7 • *deletion-type*: dump
pd_spool_cleanup_level	• *number-of-days*: 7 • *deletion-type*: all	• *number-of-days*: 7 • *deletion-type*: dump	• *number-of-days*: 7 • *deletion-type*: dump
pd_thdlock_retry_time	1000	10000	10000
pd_trn_commit_optimize	ONEPHASE	ONEPHASE	NOUSE
pd_trn_send_decision_interval	Value of pd_trn_send_decision_intval_sec operand	5	5
pd_work_buff_mode	pool	pool	each
pd_work_table_option	1	1	0
SHMMAX	200 (only in the 32-bit mode)	200 (only in the 32-bit mode)	6

(2) Operands that are no longer needed

Upgrading eliminates the need to specify the operands listed below. If you have upgraded your system and have left these operands in place, no error occurs.

Operand that is no longer needed starting in version 08-00

- `pd_dynamic_sql_object_cache`

Operands that are no longer needed starting in version 07-00

- `pd_multi_fes`
- `pd_redo_skip_inf`

Chapter

2. System Common Definition

This chapter explains the operands of the system common definition.

This chapter contains the following sections:

2.1 Operand formats

The system common definition defines the structure of the overall HiRDB system, as well as information that is common to all HiRDB system units. This section explains the formats used to specify the operands of a system common definition. Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *2.2 Operands whose specification values can be changed* and *2.3 Operand explanations*.

For users who are creating HiRDB system definitions for the first time

The first step is to determine the values to be specified for the operands shown in boldface type. In principle, HiRDB can be started once the boldface operands have been specified.

No.	Format	Operand category
1	**set pd_system_id =** ***HiRDB-identifier***	System structure
2	**[set pd_name_port =** ***HiRDB-port-number*****]**	
3	**set pd_master_file_name =** ***"HiRDB-file-name-at-beginning-of-master-directory-RDAREA"***	
4	**[set pd_max_users =** ***maximum number of concurrent connections*****]**	Maximum concurrent executions
5	**[set pd_max_server_process =** ***maximum-number-of-concurrently-activated-server-processes*****]**	
6	**[set pd_max_access_tables =** ***concurrently-accessible-base-tables-count*****]**	
7	`[set pd_utl_exec_mode = 0 | 1 ]`	
8	`[set pd_max_commit_write_reclaim_no =` *maximum-number-of-concurrent-executions-of-pdreclaim-commands-with-p-option-specified*`]`	
9	`[set pd_mode_conf = AUTO | MANUAL1 | MANUAL2]`	HiRDB startup
10	`[set pd_system_complete_wait_time =` *pdstart-command-completion-wait-time*`]`	

No.	Format	Operand category
11	[set pd_start_time_out = *HiRDB-start-preparation-maximum-wait-time*]	
12	[set pd_term_watch_count = *maximum-number-of-successive-abnormal-terminations*]	
13	[set pd_start_level = 0 | 1]	Reduced activation
14	[set pd_reduced_check_time = *wait-time-for-reduced-activation-startup-notice*]	
15	[set pd_start_skip_unit = *name-of-unit-not-to-be-started* [,*name-of-unit-not-to-be-started*]...]	
16	[set pd_dbsync_point = sync | commit]	HiRDB processing
17	[set pd_system_dbsync_point = sync | commit]	
18	[set pd_dbsync_altwrite_skip = Y | N]	
19	[set pd_overflow_suppress = Y | N]	
20	[set pd_process_terminator = resident | fixed | nonresident]	
21	[set pd_process_terminator_max = *maximum-number-of-resident-post-processing-processes*]	
22	[set pd_space_level = 0 | 1 | 3]	
23	[set pd_dec_sign_normalize = Y | N]	
24	[set pd_server_entry_queue = spnfifo | fifo | loop]	
25	[set pd_thdlock_sleep_func = 0 | 1]	
26	[set pd_thdlock_wakeup_lock = Y | N]	

No.	Format	Operand category
27	[set pd_thdlock_pipe_retry_interval = *thread-lock-release-check-interval*]	
28	[set pd_thdlock_retry_time = *thread-lock-sleep-time*]	
29	[set pd_thdspnlk_spn_count = *thread-spin-lock-spin-count*]	
30	[set pd_pageaccess_mode = SNAPSHOT | NORMAL]	
31	[set pd_cmdhold_precheck = Y | N]	
32	[set pd_db_io_error_action = dbhold | unitdown]	
33	[set pd_connect_errmsg_hide = Y | N]	
34	[set pd_cancel_down_msgchange = Y | N]	
35	[set pd_max_recover_process = *concurrently-executable-full-recovery-processes-count*]	Full recovery processing
36	[set pd_redo_allpage_put= Y | N]	
37	[set pd_trn_rerun_branch_auto_decide = Y | N]	Transaction decision processing
38	[set pd_trn_send_decision_intval_sec = *transmission-retry-interval-in-seconds-for-automatic-transaction-decision*]	
39	[set pd_trn_send_decision_interval = *transmission-retry-interval-in-minutes-for-automatic-transaction-decision*]	
40	[set pd_trn_send_decision_retry_time = *maximum-wait-time-for-transaction-auto-decision*]	
41	[set pd_trn_watch_time = *maximum-communication-wait-time-during-transaction-synchronization-point-processing*]	
42	[set pd_trn_commit_optimize = ONEPHASE | NOUSE]	

No.	Format	Operand category
43	[set pd_optimize_level = *SQL-optimization-option* [,*SQL-optimization-option*]...]	SQL optimization
44	[set pd_additional_optimize_level = *SQL-extension-optimizing-option* [,*SQL-extension-optimizing-option*]...]	
45	[set pd_hashjoin_hashing_mode = TYPE1 | TYPE2]	
46	[set pd_hash_table_size = *hash-table-size*]	
47	[set pd_work_table_option = *work-table-processing-option*]	
48	[set pd_max_list_users = *number-of-users-who-can-own-lists-concurrently*]	Narrowed retrieval
49	[set pd_max_list_count = *number-of-lists-created-per-user*]	
50	[set pd_list_initialize_timing = INITIAL | DEFER | STANDBY]	
51	[set pd_utl_exec_time = *utility-execution-monitoring-time*]	System monitoring
52	[set pd_watch_time = *SQL-maximum-execution-time*]	
53	[set pd_queue_watch_time = *message-queue-monitoring-time*]	
54	[set pd_queue_watch_timeover_action = continue | stop]	
55	[set pd_down_watch_proc = *upper-limit-for-server-process-abnormal-terminations* [, *monitoring-interval*]]	
56	[set pd_host_watch_interval = *host-to-host-monitoring-interval*]	
57	[set pd_watch_resource = MANUAL | AUTO]	
58	[set pd_max_users_wrn_pnt = *trigger-for-outputting-warning-message-related-to-number-of-connections-to-HiRDB-server* [, *trigger-for-resetting-warning-message-output-status*]]	

No.	Format	Operand category
59	`[set pd_max_access_tables_wrn_pnt =` *trigger-for-issuing-concurrently-accessible-base-tables-count-warning*`]`	
60	`[set pd_max_rdarea_no_wrn_pnt =` *trigger-for-issuing-RDAREAs-count-warning*`]`	
61	`[set pd_max_file_no_wrn_pnt =` *trigger-for-issuing-HiRDB-files-count-warning*`]`	
62	`[set pdwork_wrn_pnt =` *trigger-for-issuing-work-table-files-warning*`]`	
63	`[set pd_max_list_users_wrn_pnt =` *trigger-for-issuing-warning-about-number-of-users-who-have-created-lists*`]`	
64	`[set pd_max_list_count_wrn_pnt =` *trigger-for-issuing-warning-about-number-of-lists-created-by-a-user*`]`	
65	`[set pd_rdarea_list_no_wrn_pnt =` *trigger-for-issuing-warning-about-number-of-lists-created-at-serve*r[,*trigger-for-resetting-warning-output-status*]]	
66	`[set pd_cwaittime_wrn_pnt =` *output-condition-for-SQL-runtime-warning-information (% specification) | output-condition-for-SQL-runtime-warning-information (time specification)*`]`	SQL runtime warning output facility
67	`[set pd_cwaittime_report_dir =` *SQL-runtime-warning-information-file-output-destination-directory*`]`	
68	`[set pd_cwaittime_report_size =` *SQL-runtime-warning-information-file-maximum-size*`]`	
69	`[set pd_uap_exerror_log_use = YES | NO]`	Facility for output of extended SQL error information
70	`[set pd_uap_exerror_log_dir =` *SQL-error-report-file-storage-directory*`]`	
71	`[set pd_uap_exerror_log_size =` *SQL-error-report-file-maximum-size*`]`	
72	`[set pd_uap_exerror_log_param_size =` *maximum-data-size-of-parameter-information-to-be-output-to-error-log-file-and-SQL-error-report-file*`]`	
73	`[set pd_delete_reserved_word_file =` *SQL-reserved-word-deletion-file-name-1*`[,`*SQL-reserved-word-deletion-file-name-2*`]...]`	SQL reserved word deletion facility

No.	Format	Operand category
74	`[set pd_lck_deadlock_info = Y	N]`	Lock
75	`[set pd_lck_wait_timeout =` *lock-release-wait-time*`]`	
76	`[set pd_lck_release_detect = interval	pipe]`	
77	`[set pd_lck_release_detect_interval =` *lock-release-detection-interval*`]`	
78	`[set pd_nowait_scan_option = LOCK	NOLOCK]`	
79	`[set pd_lck_queue_limit =` *trigger-for-issuing-warning-about-number-of-users-waiting-for-lock-release*`]`	
80	`[set pd_deadlock_priority_use = Y	N]`	
81	`[set pd_command_deadlock_priority = 32	64	96	120]`	
82	`[set pd_key_resource_type = TYPE1	TYPE2]`	
83	`[set pd_indexlock_mode = {KEY	NONE}]`	
84	**[set pd_sql_object_cache_size = *SQL-object-buffer-size*]**	Buffers
85	`[set pd_def_buf_control_area_assign = INITIAL	TRAN]`	
86	`[set pd_thread_max_stack_size =` *maximum-stack-size-for-use-by-one-thread*`]`	
87	`[set pd_shmpool_attribute = free	fixed]`	Shared memory
88	`[set pd_dbbuff_attribute = free	fixed]`	

No.	Format	Operand category
89	[set pd_mlg_msg_log_unit = manager | local]	Message log files
90	[set pd_mlg_file_size = *maximum-message-log-file-size*]	
91	[set pd_statistics = Y | N]	Statistical information
92	[set pd_stj_file_size = *maximum-statistics-log-file-size*]	
93	[set pd_stj_buff_size = *statistics-log-buffer-size*]	
94	[set pd_rpc_trace = Y | N]	RPC trace information
95	[set pd_rpc_trace_name = *"name-for-RPC-trace-collection-files"*]	
96	[set pd_rpc_trace_size = *RPC-trace-collection-file-size*]	
97	[set pd_cancel_dump = put | noput]	Troubleshooting information
98	[set pd_client_waittime_over_abort = Y | N]	
99	[set pd_dump_suppress_watch_time = *troubleshooting-information-output-suppression-time*]	
100	[set pd_debug_info_netstat = Y | N]	
101	[set pd_spool_cleanup_interval = *troubleshooting-information-deletion-interval*]	
102	[set pd_spool_cleanup_interval_level = *number-of-days*[, *deletion-type*]]	
103	[set pd_spool_cleanup = normal | force | no]	
104	[set pd_spool_cleanup_level = *number-of-days* [,*deletion-type*]]	

No.	Format	Operand category
105	[set pd_module_trace_max = *maximum-number-of-module-traces-that-can-be-stored*]	
106	[set pd_module_trace_timer_level = *module-trace-output-time-acquisition-method*]	
107	**[set pd_max_rdarea_no =** *maximum-number-of-RDAREAs***]**	RDAREAs
108	**[set pd_max_file_no =** *maximum-number-of-HiRDB-files-comprising-an-RDAREA***]**	
109	[set pd_rdarea_warning_point = *segment-usage-ratio-1* [,*segment-usage-ratio-2* [,*segment-usage-ratio-3*]]]	
110	[set pd_rdarea_open_attribute_use = Y | N]	
111	[set pd_rdarea_open_attribute = INITIAL | DEFER | SCHEDULE]	
112	[set pd_shared_rdarea_use = Y | N]	
113	[set pd_dbbuff_lru_option = SEPARATE | MIX]	Global buffers
114	[set pd_dbbuff_modify = Y | N]	
115	[set pd_dbbuff_lock_release_detect = pipe | interval]	
116	[set pd_dbbuff_lock_spn_count = *number-of-spins-during-lock-acquisition-wait-processing*]	
117	[set pd_dbbuff_lock_interval = *interval-during-lock-acquisition-wait-processing*]	
118	[set pd_dbbuff_wait_interval = *global-buffer-occupation-state-check-interval*]	
119	[set pd_dbbuff_wait_spn_count = *maximum-spin-loop-count-for-global-buffer-occupation-state-checking*]	
120	[set pd_dbbuff_rate_updpage = *deferred-write-trigger-request-rate*]	

No.	Format	Operand category
121	[set pd_assurance_table_no = *minimum-guaranteed-tables-count*]	Table or index reservation count
122	[set pd_assurance_index_no = *minimum-guaranteed-indexes-count*]	
123	[set pd_constraint_name = LEADING | TRAILING]	Referential and check constraints
124	[set pd_check_pending = USE | NOUSE]	
125	[set pd_large_file_use = Y | N]	HiRDB file system areas
126	[set pd_rorg_predict = Y | N]	Facility for predicting reorganization time
127	[set pd_audit = Y | N]	Security audit facility
128	[set pd_aud_file_name = *HiRDB-file-system-area-name-for-audit-trail-file*]	
129	[set pd_aud_max_generation_size = *audit-trail-file-maximum-size*]	
130	[set pd_aud_max_generation_num = *maximum-audit-trail-file-count*]	
131	[set pd_aud_no_standby_file_opr = down | forcewrite]	
132	[set pd_aud_async_buff_size = *size-of-buffer-used-for-asynchronous-output-of-audit-trail-file*]	
133	[set pd_aud_async_buff_count = *number-of-buffer-sectors-used-for-asynchronous-output-of-audit-trail-file*]	
134	[set pd_aud_async_buff_retry_intvl = *retry-interval-for-allocation-of-a-buffer-to-be-used-for-asynchronous-output-of-audit-trail-file*]	
135	[set pd_aud_file_wrn_pnt = *warning-message-output-trigger*[,*trigger-for-resetting-warning-message-output-status*]]	

No.	Format	Operand category
136	[set pd_ha = use | nouse]	System switchover facility
137	[set pd_ha_ipaddr_inherit = Y | N]	
138	[set pd_ha_switch_timeout = Y | N]	
139	[set pd_ha_prc_cleanup_check = Y | N]	
140	[set pd_ha_mgr_rerun = wait | notwait]	
141	[set pd_ha_transaction = error | queuing]	
142	[set pd_ha_trn_queuing_wait_time = *transaction-queuing-wait-time*]	
143	[set pd_ha_trn_restart_retry_time = *retry-time-upper-limit-after-transaction-start-request-errors*]	
144	[set pd_ha_resource_act_wait_time = *maximum-wait-time-for-resource-activation*]	
145	[set pd_rpl_init_start = Y | N]	HiRDB Datareplicator
146	[set pd_rpl_reflect_mode = server | uap]	
147	[set pd_log_rpl_no_standby_file_opr = stop | continue]	
148	[set pd_jp1_use = Y | N]	Linkage to JP1
149	[set pd_jp1_event_level = 1 | 2]	
150	[set pd_jp1_event_msg_out = Y | N]	
151	[set pd_directory_server = sods]	Directory Server linkage facility

No.	Format	Operand category
152	[set pd_rise_use = Y | N]	Real Time SAN Replication
153	[set pd_rise_pairvolume_combination = sync | async | hybrid| syssync]	
154	[set pd_rise_disaster_mode = normal | alone]	
155	[set pd_rise_fence_level = data | never]	
156	[set pd_inner_replica_control = *inner-replica-maximum-group-count*]	Inner replica facility
157	[set pd_lv_mirror_use = Y | N]	
158	[set pd_max_reflect_process_count = *number-of-processes-to-be-allocated-for-reflection-processing*]	
159	[set pd_log_org_reflected_logpoint = keep | release]	
160	[set pd_log_org_no_standby_file_opr = stop | continue]	
161	[set pd_max_foreign_server = *foreign-server-maximum-count*]	HiRDB External Data Access facility
162	[set pd_oltp_holdcr = use | nouse]	OLTP
163	[set pd_auto_vrup = Y | N]	Version upgrade
164	[set pd_sysdef_default_option = recommendable | v6compatible | v7compatible]	
165	[set pd_service_port = *port-number-for-high-speed-connection-from-client*]	Communication processing
166	[set pd_change_clt_ipaddr = 0 | 1]	
167	[set pd_registered_port = *"port-number-reservation-range"* [,*"port-number-reservation-range"*]...]	

No.	Format	Operand category
168	[set pd_registered_port_check = Y | N | C | W]	
169	[set pd_registered_port_level = 0 | 1]	
170	[set pd_ipc_send_retrycount = *process-to-process-send-retries-count*]	
171	[set pd_ipc_send_retrysleeptime = *process-to-process-send-retry-sleep-time*]	
172	[set pd_ipc_send_count = *server-to-server-send-retries-count*]	
173	[set pd_ipc_recv_count = *server-to-server-receive-retries-count*]	
174	[set pd_ipc_conn_nblock = Y | N]	
175	[set pd_ipc_conn_nblock_time = *connection-establishment-monitoring-time-in-non-block-mode*]	
176	[set pd_ipc_conn_interval = *connection-establishment-retry-interval*]	
177	[set pd_ipc_conn_count = *connection-establishment-retry-count*]	
178	[set pd_ipc_inet_bufsize = *send-receive-buffer-size-for-server-unit-to-unit-communication*]	
179	[set pd_ipc_unix_bufsize = *send-receive-buffer-size-for-communication-within-server-unit*]	
180	[set pd_tcp_inet_bufsize = *send-receive-buffer-size-for-communication-with-HiRDB-client-outside-host-where-HiRDB-server-resides*]	
181	[set pd_tcp_unix_bufsize = *send-receive-buffer-size-for-communication-with-HiRDB-client-inside-host-where-HiRDB-server-resides*]	
182	[set pd_utl_buff_size = *utility-communication-buffer-size*]	
183	[set pd_sql_send_buff_size = *buffer-size-for-inter-server-communication-for-SQL-execution*]	

No.	Format	Operand category
184	[set pd_ipc_tcp_nodelayack = Y | N]	
185	[set pd_java_option = *"Java-option"* [,*"Java-option"*]...]	Java
186	[set pd_java_routine_stack_size = *stack-size-for-use-by-Java-routine*]	
187	[set pd_java_archive_directory = *"JAR-file-storage-directory"*]	
188	[set pd_java_classpath = *"Java-class-path"*]	
189	[set pd_java_runtimepath = *"Java-Runtime-Environment-root-directory"*]	
190	[set pd_java_libpath = *"Java-virtual-machine-library-directory"*]	
191	[set pd_java_stdout_file = *"Java-virtual-machine-standard-output-and-standard-error-output-destination-file"*]	
192	**{{ pdunit -x *host-name*** **-u *unit-identifier*** **[-d "*HiRDB-directory-name*"]** **[-c *host-name*]** **[-p *port-number*]}}**	Unit structure
193	**{{ pdstart -t *server-type*** **[-s *server-name*]** **-x *host-name* | -u *unit-identifier*** **[-m *host-name*[,*host-name*]...** [-n ***host-name*** [, ***host-name***] **...]]** **[-c *server-name* | -g *HA-group-identifier*]** **[-k stls]}}**	Server structure
194	**[{{ pdbuffer -a *global buffer-name*** **{-r *RDAREA-name*[,*RDAREA-name*]...** | **-b *RDAREA-name* [,*RDAREA-name*]...** | **-o** | **-i *authorization-identifier.index-identifier*}** **-n *buffer-sectors-count*[-l *buffer-size*]** [-m *maximum-concurrently-executable-prefetches-count*] [-p *maximum-batch-input-pages-count*] [-w *updated-pages-output-rate-during-deferred-write-trigger*] [-c] [-y *update-buffer-sectors-count-for-deferred-write-trigger-event*]}}]	Global buffers

No.	Format	Operand category
195	[pdhagroup -g *HA-group-identifier* -u *unit-identifier*[,*unit-identifier*]...]	HA groups
196	[pdstbegin[-k *statistical-information-type*[,*statistical-information-type*]...] [-m *interval*] [{-x *host-name* | -u *unit-identifier* }] [{ -a | -s *server-name*[, *server-name*]...}]]	Statistical information
197	[pdhibegin -k *statistics-type*[,*statistics-type*]...]	
198	{{[pdcltgrp -g *client-group-name* -u *guaranteed-number-of-connected-users-per-group*]}}	Client group
199	{{[pdplugin -n *plug-in-name*]}}	Plug-ins
200	[putenv HORCMINST *RAID-Manager-instance-number*]	Real Time SAN Replication
201	**[putenv SHMMAX *maximum-shared-memory-segment-size*]**	Shared memory
202	[putenv TZ *time-zone*]	Date and time
203	[pdmlgput -s *output-selection* {-c ALL | -m *message-ID*[,*message-ID*]...}]	Message output suppression facility
204	[set pd_substr_length = 3 | 4 | 5 | 6]	Character encoding

2.2 Operands whose specification values can be changed

The values of some system common definition operands can be changed in the unit control information definition and in the individual server definitions. These operands are indicated below. After a planned termination, forced termination, or abnormal termination of HiRDB, some HiRDB system definition operands can be modified while others cannot be modified. The operands that can be modified are indicated below.

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
1	`pd_system_id`	Y	—	—	—	—	—	—	N	N
2	`pd_name_port`	Y	—	—	—	—	—	—	Y	Y
3	`pd_master_file_name`	Y	—	—	—	—	—	—	N	N
4	`pd_max_users`	Y	—	—	—	—	—	—	N	Y
5	`pd_max_server_process`	Y	Y	—	—	—	—	—	Y	Y
6	`pd_max_access_tables`	Y	—	—	—	—	—	—	Y	Y
7	`pd_utl_exec_mode`	Y	—	—	—	—	—	—	N	Y
8	`pd_max_commit_write_reclaim_no`	Y	—	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
9	pd_mode_conf	Y	—	—	—	—	—	—	Y	Y
10	pd_system_complete_wait_time	Y	—	—	—	—	—	—	Y	Y
11	pd_start_time_out	Y	—	—	—	—	—	—	Y	Y
12	pd_term_watch_count	Y	Y	—	—	—	—	—	Y	Y
13	pd_start_level	Y	—	—	—	—	—	—	Y	Y
14	pd_reduced_check_time	Y	—	—	—	—	—	—	Y	Y
15	pd_start_skip_unit	Y	—	—	—	—	—	—	Y	Y
16	pd_start_skip_unit	Y	—	—	—	—	—	—	Y	Y
17	pd_dbsync_point	Y	—	—	—	—	—	—	N	Y
18	pd_system_dbsync_point	Y	—	—	—	—	—	—	N	Y
19	pd_overflow_suppress	Y	—	—	—	—	—	—	Y	Y
20	pd_process_terminator	Y	—	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
21	`pd_process_terminator_max`	Y	—	—	—	—	—	—	Y	Y
22	`pd_space_level`[5]	Y	—	—	—	—	—	—	Y	Y
23	`pd_dec_sign_normalize`	Y	—	—	—	—	—	—	N	Y
24	`pd_server_entry_queue`	Y	Y	—	—	—	—	—	Y	Y
25	`pd_thdlock_sleep_func`	Y	—	—	—	—	—	—	Y	Y
26	`pd_thdlock_wakeup_lock`	Y	Y	—	—	—	—	—	Y	Y
27	`pd_thdlock_pipe_retry_interval`	Y	Y	—	—	—	—	—	Y	Y
28	`pd_thdlock_retry_time`	Y	Y	—	—	—	—	—	Y	Y
29	`pd_thdspnlk_spn_count`	Y	Y	—	—	—	—	—	Y	Y
30	`pd_pageaccess_mode`	Y	—	—	—	—	—	—	Y	Y
31	`pd_cmdhold_precheck`	Y	—	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
32	pd_db_io_error_action	Y	Y	—	—	—	—	—	Y	Y
33	pd_connect_errmsg_hide	Y	—	—	—	—	—	—	Y	Y
34	pd_cancel_down_msgchange	Y	—	—	—	—	—	—	Y	Y
35	pd_max_recover_process	Y	Y	—	—	—	—	—	Y	Y
36	pd_redo_allpage_put	Y	—	—	—	—	—	—	Y	Y
37	pd_trn_rerun_branch_auto_decide	Y	—	—	—	—	—	—	Y	Y
38	pd_trn_send_decision_intval_sec	Y	—	—	—	—	—	—	Y	Y
39	pd_trn_send_decision_interval	Y	—	—	—	—	—	—	Y	Y
40	pd_trn_send_decision_retry_time	Y	—	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
41	pd_trn_watch_time	Y	—	—	—	—	—	—	Y	Y
42	pd_trn_commit_optimize	Y	—	—	—	—	—	—	Y	Y
43	pd_optimize_level[5]	Y	—	—	—	Y	—	—	Y	Y
44	pd_additional_optimize_level[5]	Y	—	—	—	Y	—	—	Y	Y
45	pd_hashjoin_hashing_mode[5]	Y	—	—	—	—	—	—	Y	Y
46	pd_hash_table_size[5]	Y	—	—	—	—	—	—	N	N
47	pd_work_table_option	Y	—	—	—	—	—	—	Y	Y
48	pd_max_list_users	Y	—	—	—	—	—	—	N[2]	Y
49	pd_max_list_count	Y	—	—	—	—	—	—	N	Y
50	pd_list_initialize_timing	Y	—	—	—	—	—	—	Y	Y
51	pd_utl_exec_time	Y	—	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
52	pd_watch_time	Y	Y	—	—	—	—	—	Y	Y
53	pd_queue_watch_time	Y	—	—	—	—	—	—	Y	Y
54	pd_queue_watch_timeover_action	Y	—	—	—	—	—	—	Y	Y
55	pd_down_watch_proc	Y	Y	—	—	—	—	—	Y	Y
56	pd_host_watch_interval	Y	—	—	—	—	—	—	Y	Y
57	pd_watch_resource	Y	—	—	—	—	—	—	Y	Y
58	pd_max_users_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
59	pd_max_access_tables_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
60	pd_max_rdarea_no_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
61	pd_max_file_no_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
62	pdwork_wrn_pnt	Y	—	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
63	pd_max_list_users_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
64	pd_max_list_count_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
65	pd_rdarea_list_no_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
66	pd_cwaittime_wrn_pnt[5]	Y	—	—	Y	Y	—	—	Y	Y
67	pd_cwaittime_report_dir	Y	Y	—	—	—	—	—	Y	Y
68	pd_cwaittime_report_size	Y	Y	—	—	—	—	—	Y	Y
69	pd_uap_exerror_log_use[5]	Y	—	—	—	—	—	—	Y	Y
70	pd_uap_exerror_log_dir	Y	Y	—	—	—	—	—	Y	Y
71	pd_uap_exerror_log_size	Y	Y	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
72	`pd_uap_exerror_log_param_size`[5]	Y	Y	—	—	—	—	—	Y	Y
73	`pd_delete_reserved_word_file`[5]	Y	—	—	—	—	—	—	Y	Y
74	`pd_lck_deadlock_info`	Y	Y	—	—	—	—	—	Y	Y
75	`pd_lck_wait_timeout`	Y	Y	—	—	—	—	—	Y	Y
76	`pd_lck_release_detect`	Y	Y	—	—	—	—	—	Y	Y
77	`pd_lck_release_detect_interval`	Y	Y	—	—	—	—	—	Y	Y
78	`pd_nowait_scan_option`	Y	—	—	—	—	—	—	Y	Y
79	`pd_lck_queue_limit`	Y	—	—	—	—	—	—	Y	Y
80	`pd_deadlock_priority_use`	Y	—	—	—	—	—	—	Y	Y
81	`pd_command_deadlock_priority`	Y	—	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
82	pd_key_resource_type	Y	—	—	—	—	—	—	N	Y
83	pd_indexlock_mode	Y	—	—	—	—	—	—	N	Y
84	pd_sql_object_cache_size	Y	—	Y	Y	Y	Y	Y	N	Y
85	pd_def_buf_control_area_assign	Y	—	—	—	—	—	—	Y	Y
86	pd_thread_max_stack_size	Y	—	—	—	—	—	—	Y	Y
87	pd_shmpool_attribute	Y	Y	—	—	—	—	—	Y	Y
88	pd_dbbuff_attribute	Y	Y	—	—	—	—	—	Y	Y
89	pd_mlg_msg_log_unit	Y	—	—	—	—	—	—	N	N
90	pd_mlg_file_size	Y	—	—	—	—	—	—	Y	Y
91	pd_statistics	Y	—	—	—	—	—	—	Y	Y
92	pd_stj_file_size	Y	Y	—	—	—	—	—	N[2]	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
93	pd_stj_buff_size	Y	Y	—	—	—	—	—	N2	Y
94	pd_rpc_trace	Y	Y	Y	Y	Y	Y	Y	Y	Y
95	pd_rpc_trace_name	Y	Y	Y	Y	Y	Y	Y	Y	Y
96	pd_rpc_trace_size	Y	Y	Y	Y	Y	Y	Y	Y	Y
97	pd_cancel_dump	Y	Y	—	—	—	—	—	Y	Y
98	pd_client_waittime_over_abort	Y	—	—	—	—	—	—	Y	Y
99	pd_dump_suppress_watch_time	Y	Y	—	—	—	—	—	Y	Y
100	pd_debug_info_netstat	Y	—	—	—	—	—	—	Y	Y
101	pd_spool_cleanup_interval	Y	Y	—	—	—	—	—	Y	Y
102	pd_spool_cleanup_interval_level	Y	Y	—	—	—	—	—	Y	Y
103	pd_spool_cleanup	Y	Y	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
104	`pd_spool_cleanup_level`	Y	Y	—	—	—	—	—	Y	Y
105	`pd_module_trace_max`	Y	Y	Y	Y	Y	Y	Y	Y	Y
106	`pd_module_trace_timer_level`	Y	Y	Y	Y	Y	Y	Y	Y	Y
107	`pd_max_rdarea_no`	Y	—	—	—	—	—	—	N	Y
108	`pd_max_file_no`	Y	—	—	—	—	—	—	Y[1,2]	Y
109	`pd_rdarea_warning_point`	Y	—	—	—	—	—	—	Y	Y
110	`pd_rdarea_open_attribute_use`	Y	—	—	—	—	—	—	Y	Y
111	`pd_rdarea_open_attribute`	Y	—	—	—	—	—	—	Y	Y
112	`pd_shared_rdarea_use`	Y	—	—	—	—	—	—	N	Y
113	`pd_dbbuff_lru_option`	Y	—	—	—	—	—	—	Y	Y
114	`pd_dbbuff_modify`	Y	—	—	—	—	—	—	N	N

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
115	pd_dbbuff_lock_release_detect	Y	—	—	—	—	—	—	Y	Y
116	pd_dbbuff_lock_spn_count	Y	—	—	—	—	—	—	Y	Y
117	pd_dbbuff_lock_interval	Y	—	—	—	—	—	—	Y	Y
118	pd_dbbuff_wait_interval	Y	Y	—	—	—	—	—	Y	Y
119	pd_dbbuff_wait_spn_count	Y	Y	—	—	—	—	—	Y	Y
120	pd_dbbuff_rate_updpage	Y	—	—	—	—	—	—	Y	Y
121	pd_assurance_table_no	Y	—	—	—	—	—	—	Y[3]	Y
122	pd_assurance_index_no	Y	—	—	—	—	—	—	Y[3]	Y
123	pd_constraint_name	Y	—	—	—	—	—	—	Y	Y
124	pd_check_pending	Y	—	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
125	pd_large_file_use	Y	—	—	—	—	—	—	N	Y
126	pd_rorg_predict	Y	—	—	—	—	—	—	Y	Y
127	pd_audit	Y	Y	—	—	—	—	—	N	N
128	pd_aud_file_name	Y	Y	—	—	—	—	—	Y	Y
129	pd_aud_max_generation_size	Y	Y	—	—	—	—	—	Y	Y
130	pd_aud_max_generation_num	Y	Y	—	—	—	—	—	Y	Y
131	pd_aud_no_standby_file_opr	Y	—	—	—	—	—	—	Y	Y
132	pd_aud_async_buff_size	Y	Y	—	—	—	—	—	Y	Y
133	pd_aud_async_buff_count	Y	Y	—	—	—	—	—	Y	Y
134	pd_aud_async_buff_retry_intvl	Y	Y	—	—	—	—	—	Y	Y
135	pd_aud_file_wrn_pnt	Y	—	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
136	pd_ha	Y	—	—	—	—	—	—	N	N
137	pd_ha_ipaddr_inherit	Y	Y	—	—	—	—	—	N	N
138	pd_ha_switch_timeout	Y	Y	—	—	—	—	—	Y	Y
139	pd_ha_prc_cleanup_check	Y	—	—	—	—	—	—	Y	Y
140	pd_ha_mgr_rerun	Y	—	—	—	—	—	—	Y	Y
141	pd_ha_transaction[5]	Y	—	—	—	—	—	—	Y	Y
142	pd_ha_trn_queuing_wait_time	Y	—	—	—	—	—	—	Y	Y
143	pd_ha_trn_restart_retry_time	Y	—	—	—	—	—	—	Y	Y
144	pd_ha_resource_act_wait_time	Y	Y	—	—	—	—	—	Y	Y
145	pd_rpl_init_start	Y	—	—	—	—	—	—	N	N
146	pd_rpl_reflect_mode	Y	—	—	—	—	—	—	N	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
147	pd_log_rpl_no_standby_file_opr	Y	--	--	--	--	--	--	Y	Y
148	pd_jp1_use	Y	--	--	--	--	--	--	N	N
149	pd_jp1_event_level	Y	--	--	--	--	--	--	Y	Y
150	pd_jp1_event_msg_out	Y	--	--	--	--	--	--	Y	Y
151	pd_directory_server	Y	--	--	--	--	--	--	Y	Y
152	pd_rise_use	Y	--	--	--	--	--	--	N	N
153	pd_rise_pairvolume_combination	Y	--	--	--	--	--	--	N	N
154	pd_rise_disaster_mode	Y	--	--	--	--	--	--	Y	Y
155	pd_rise_fence_level	Y	--	--	--	--	--	--	Y	Y
156	pd_inner_replica_control	Y	--	--	--	--	--	--	N	Y
157	pd_lv_mirror_use	Y	--	--	--	--	--	--	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
158	`pd_max_reflect_process_count`	Y	—	—	—	—	—	—	N	N
159	`pd_log_org_reflected_logpoint`	Y	—	—	—	—	—	—	Y[4]	Y[4]
160	`pd_log_org_no_standby_file_opr`	Y	—	—	—	—	—	—	Y	Y
161	`pd_max_foreign_server`	Y	—	—	—	—	—	—	N	Y
162	`pd_oltp_holdcr`	Y	—	—	—	—	—	—	Y	Y
163	`pd_auto_vrup`	Y	—	—	—	—	—	—	N	N
164	`pd_sysdef_default_option`	Y	—	—	—	—	—	—	N	Y
165	`pd_service_port`	Y	Y	—	—	—	—	—	Y	Y
166	`pd_change_clt_ipaddr`	Y	Y	—	—	—	—	—	Y	Y
167	`pd_registered_port`	Y	Y	—	—	—	—	—	N	Y
168	`pd_registered_port_check`	Y	Y	—	—	—	—	—	N	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
169	pd_registered_port_level	Y	Y	—	—	—	—	—	N	Y
170	pd_ipc_send_retrycount	Y	Y	—	—	—	—	—	Y	Y
171	pd_ipc_send_retrysleeptime	Y	Y	—	—	—	—	—	Y	Y
172	pd_ipc_send_count	Y	Y	—	—	—	—	—	Y	Y
173	pd_ipc_recv_count	Y	Y	—	—	—	—	—	Y	Y
174	pd_ipc_conn_nblock	Y	—	—	—	—	—	—	Y	Y
175	pd_ipc_conn_nblock_time	Y	—	—	—	—	—	—	Y	Y
176	pd_ipc_conn_interval	Y	—	—	—	—	—	—	Y	Y
177	pd_ipc_conn_count	Y	—	—	—	—	—	—	Y	Y
178	pd_ipc_inet_bufsize	Y	Y	—	—	—	—	—	Y	Y
179	pd_ipc_unix_bufsize	Y	Y	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	SDS	FES	DS	BES		
180	pd_tcp_inet_bufsize	Y	Y	—	—	—	—	—	Y	Y
181	pd_tcp_unix_bufsize	Y	Y	—	—	—	—	—	Y	Y
182	pd_utl_buff_size	Y	—	—	—	—	—	—	Y[1]	Y[1]
183	pd_sql_send_buff_size	Y	—	—	—	—	—	—	Y	Y
184	pd_ipc_tcp_nodelayack	Y	—	—	—	—	—	—	Y	Y
185	pd_java_option	Y	—	—	—	—	—	—	Y	Y
186	pd_java_routine_stack_size	Y	—	—	—	—	—	—	Y	Y
187	pd_java_archive_directory	Y	Y	—	—	—	—	—	Y	Y
188	pd_java_classpath	Y	Y	—	—	—	—	—	Y	Y
189	pd_java_runtimepath	Y	Y	—	—	—	—	—	Y	Y
190	pd_java_libpath	Y	Y	—	—	—	—	—	Y	Y
191	pd_java_stdout_file	Y	Y	Y	Y	Y	Y	Y	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
192	pdunit	Y	—	—	—	—	—	—	N	N
193	pdstart	Y	—	—	—	—	—	—	N	N
194	pdbuffer	Y	—	—	—	—	—	—	N	Y
195	pdhagroup	Y	—	—	—	—	—	—	N	N
196	pdstbegin	Y	—	—	—	—	—	—	Y	Y
197	pdhibegin	Y	—	—	—	—	—	—	Y	Y
198	pdcltgrp	Y	—	—	—	—	—	—	Y	Y
199	pdplugin	Y	—	—	—	—	—	—	N	N
200	HORCMINST	Y	—	—	—	—	—	—	Y	Y
201	SHMMAX	Y	Y	—	—	—	—	—	Y	Y
202	TZ	Y	—	—	—	—	—	—	Y	Y
203	pdmlgput	Y	—	—	—	—	—	—	Y	Y
204	pd_substr_length	Y	—	—	—	—	—	—	Y	Y

Y: Yes, specification value can be modified.

N: No, specification valued cannot be modified.

—: Specification value cannot be modified because the operand is not applicable.

SYS: System common definition

UNT: Unit control information definition

SVR: Server common definition

SDS: Single server definition

FES: Front-end server definition

DS: Dictionary server definition

BES: Back-end server definition

[1] If the specified value is too small, it may not be possible to restart HiRDB. When this is the case, restore the value to what it was before the change and then restart HiRDB.

[2] If the specified value is too large, it may not be possible to restart HiRDB. When this is the case, restore the value to what it was before the change and then restart HiRDB.

[3] If `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand, the value before the change is used to start HiRDB even if the definition is changed during the HiRDB restart.

[4] Some restrictions apply to the operand values when you are modifying them. For details, see the explanation of operands.

[5] The values of the following HiRDB system definition operands can be changed for each client in its client environment definition. To change values for a client, you must specify the applicable operands in its client environment definition. For details about the client environment definition, see the *HiRDB Version 8 UAP Development Guide*.

HiRDB system definition operand	Client environment definition operand
`pd_additional_optimize_level`	`PDADDITIONALOPTLVL`
`pd_cwaittime_wrn_pnt`	`PDCWAITTIMEWRNPNT`
`pd_delete_reserved_word_file`	`PDDELRSVWDFILE`
`pd_hash_table_size`	`PDHASHTBLSIZE`
`pd_hashjoin_hashing_mode`	`PDHJHASHINGMODE`
`pd_optimize_level`	`PDSQLOPTLVL`
`pd_space_level`	`PDSPACELVL`
`pd_ha_transaction`	`PDHATRNQUEUING`
`pd_uap_exerror_log_use`	`PDUAPEXERLOGUSE`
`pd_uap_exerror_log_param_size`	`PDUAPEXERLOGPRMSZ`

2.3 Operand explanations

2.3.1 Operands related to the system structure

1) pd_system_id = *HiRDB-identifier*

～ <identifier>((4 alphabetics))

Specifies an identifier for the HiRDB system. This operand is required.

In the following cases, a unique HiRDB identifier must be used in each system:

- Multiple HiRDB/Single Servers are being used
- A utility special unit is being used

Notes

The only way to change a HiRDB identifier after it has been set is to use the database initialization utility to rebuild the system. For this reason, you should avoid a name that may have to be changed later.

2) pd_name_port = *HiRDB-port-number*

～ <unsigned integer> ((5001-65535)) <<20000>>

Specifies the port number that becomes the HiRDB system address.

Specification guidelines

- This port number must not be the same as the port number used by any other program; no other program may use the port number specified here.
- To start multiple HiRDB servers or units on the same machine, a different port number must be specified for each HiRDB.

Notes

- You must specify a number that is outside the range of port numbers assigned automatically by the OS. The range of port numbers assigned automatically by the OS depends on the OS.
- If a port number is specified that is within the range of port numbers assigned automatically by the OS, that number may be in use by another program. If so, the HiRDB server does not start.

3) pd_master_file_name = "*HiRDB-file-name-at-beginning-of-master-directory-RDAREA*"

～ <pathname>((up to 167 characters))

Specifies as an absolute pathname the name of the HiRDB file that is at the beginning of the master directory RDAREA.

This operand is required.

Notes

- This operand may not be specified for a utility special unit.
- Once the name of the HiRDB file that is at the beginning of the master directory RDAREA is specified, it cannot be changed. To change it, you must rebuild the system with the database initialization utility. For this reason, you should avoid a name that may have to be changed later.

2.3.2 Operands related to maximum concurrent executions

4) pd_max_users = *maximum number of concurrent connections*

~ <unsigned integer>((1-2000))<<10>>

This operand specifies the maximum number of concurrent connections allowed for the HiRDB server. For a HiRDB/Parallel Server, this operand specifies the maximum number of concurrent connections allowed for a single front-end server. If the number of connection requests exceeds the value specified by this operand, HiRDB rejects any additional connection requests (connection requests result in errors). Note that connection in this case means an attempt to connect to a HiRDB server based on a CONNECT statement.

Specification guidelines

- When OpenTP1 is used, the connection count is the number of OpenTP1 server processes to be connected to the HiRDB server (including XA connections).
- When DBPARTNER/Server is used, the connection count is the number of DBPARTNER/Server connection clients.
- When HiRDB SQL Executer is used, the connection count is the number of HiRDB SQL Executer connection clients.
- If the multi-connection facility is used, the connection count is the total number of concurrent connections by individual UAPs.
- Some of the HiRDB commands and utilities are internally connected to the HiRDB server. Therefore, while these commands and utilities are being executed, the number of users that can be connected temporarily decreases. Determine the operand value by taking this fact into consideration. For details about command and utility connection count, see *Maximum number of concurrent executions for commands* in the manual *HiRDB Version 8 Command Reference*.

Notes

- If you increase the value of the `pd_max_users` operand, you must also increase the value of the `pd_max_server_process` operand. Along with these increases, both the size of the shared memory and the number of ports used by the HiRDB system also increase. For notes related to an increase in the value of the `pd_max_users` operand (when the number of users is increased), see the *HiRDB Version 8 System Operation Guide*. If you omit the `pd_max_server_process` operand, HiRDB automatically computes a value for this operand.
- For a multiple front-end server, the maximum number of connection requests that can be processed is obtained by multiplying the value of the `pd_max_users` operand by the number of front-end servers. However, make sure that the value obtained does not exceed the specifiable upper limit for the `pd_max_users` operand (2000).
- Depending on the value of the `pd_max_reflect_process_count` operand and the total number of reflection processing processes, the maximum number of connections may fall short of the value of the `pd_max_users` operand. For details, see the explanation of the `pd_max_reflect_process_count` operand.

5) pd_max_server_process = ***maximum-number-of-concurrently-activated-server-processes***

～ <unsigned integer>((50-10000))

Specifies the maximum number of processes that can be activated at the same time in the same unit. The number of server processes includes the number of processes for the system server, individual servers, utilities, etc. (the system server is a server that is used internally by HiRDB).

Specification guidelines

- Normally, omit this operand. When it is omitted, the value obtained from the following formula is assumed. HiRDB automatically re-computes the value of this operand if you change any of the operands mentioned in the explanation of a variable.

  Default value = $a + b \times (c + 26) + 73 + i + j + k + m + n$

- If you choose to specify this operand, use the following formula as a reference for determining the operand value. For a HiRDB/Parallel Server, determine values for all units, and use the largest of these values as a guideline.

Recommended value = $a + b \times (c + d \times e + f) + d \times g + h + i + j + k + m + n$

Variable	Explanation of variable
a	For HiRDB/Single Server: Value of pd_max_users + pd_max_reflect_process_count For HiRDB/Parallel Server: Total the values obtained using the following formulas for individual servers inside the unit: Back-end server: Value of pd_max_bes_process + pd_max_reflect_process_count Dictionary server: Value of pd_max_dic_process + pd_max_reflect_process_count Front-end server: Value of pd_max_users + pd_max_reflect_process_count • If the pd_max_bes_process or pd_max_dic_process operand is omitted, use the value of the pd_max_users operand. • If a unit contains multiple back-end servers, compute a value for each back-end server. • For a unit to which the standby-less system switchover (effects distributed) facility is to be applied, substitute the value of the pd_ha_max_server_process operand.
b	For HiRDB/Single Server: 1 For HiRDB/Parallel Server: Server count inside the unit (server count allocated to the unit by the pdstart operand of the system common definition) • For a unit to which the standby-less system switchover (effects distributed) facility is to be applied, substitute the value of the pd_ha_max_act_guest_servers operand.
c	Concurrently executable process count in full recovery processing (value of the pd_max_recover_process operand)
d	Maximum number of concurrently executable utilities ("2" is the default value for this operand)
e	Number of processes to be started in each server by the utility (set this variable to 10)
f	Number of processes to be started by HiRDB for controlling the server (set this variable to 6)
g	Number of processes to be started in each unit by the utility (set this variable to 10)
h	Number of processes to be started by HiRDB for controlling the unit (set this variable to 50)
i	For HiRDB/Single Server: 1 For HiRDB/Parallel Server: Number of back-end servers inside the unit • For a unit to which the standby-less system switchover (effects distributed) facility is to be applied, substitute the value of the pd_ha_max_act_guest_servers operand.

Variable	Explanation of variable
j	For HiRDB/Single Server: value of the `pd_max_ard_process` operand For HiRDB/Parallel Server: *number-of-back-end-servers-inside-the-unit* × value specified for the `pd_max_ard_process` operand + *number-of-dictionary-servers-inside-the-unit* × value specified for the `pd_max_ard_process` operand • For a unit to which the standby-less system switchover (effects distributed) facility is to be applied, add the following value: *maximum-value-of-pd_max_ard_process-operand-for-guest-BES* × value of the `pd_ha_max_act_guest_servers` operand
k	Depends on the value specified for the `pd_process_terminator` operand. • `resident`: 1 • `fixed` (default value): `pd_process_terminator_max` operand value • `nonresident`: 0
m	To execute updatable online reorganization, substitute the following values. If updatable online reorganization is not to be executed, substitute 0. For HiRDB/Single Server: *x* For HiRDB/Parallel Server: *x* × (*y* + *z*) *x*: `pd_max_reflect_process_count` operand value *y*: *number-of-servers-within-the-unit* *z*: `pd_ha_max_act_guest_servers` operand value For a unit to which the standby-less system switchover (effects distributed) facility is to be applied, add *z*. For a unit to which the standby-less system switchover (effects distributed) facility is not applied, the value is 0.
n	For HiRDB/Single Server: `pd_dfw_awt_process` operand value For HiRDB/Parallel Server: *number-of-back-end-servers* × `pd_dfw_awt_process` operand value + *number-of-dictionary-servers* × `pd_dfw_awt_process` operand value • For a unit to which the standby-less system switchover (effects distributed) facility is to be applied, add the following value: *maximum-value-of-pd_dfw_awt_process-operand-value-for-guest-BES* × `pd_ha_max_act_guest_servers` operand value

Relationship to other operands

- This operand's value limits the maximum value of the `pd_process_count` operand.
- If you need to adjust the value for each unit because the server configuration in each unit is different, specify the `pd_max_server_process` operand in the unit control information

definition.

Notes

- This specification value includes the number of processes for servers and utilities in the unit. If this value is too small, the following may occur:
  - Unit or server startup process results in an error
  - Transaction recovery cannot be performed
  - HiRDB planned termination cannot be performed
- Because the number of processes that can actually be activated depends on factors such as the system resources, it may be necessary to adjust the resources or to change the locations of servers in some cases.

6) pd_max_access_tables = *concurrently-accessible-base-tables-count*

～ <unsigned integer>((4-32000))<<64>>

Specifies the maximum number of tables that can be accessed concurrently in a single transaction. The number of tables that can be accessed concurrently means the total number of tables coded in the SQL statements in a transaction. If the number specified by this operand is exceeded, an error results, processing returns to the UAP, and rollback processing is executed.

Specification guidelines

- If the same table is specified in different SQLs, the total number of times the table is used must be counted.
- If the same table is specified more than once in the same SQL, the total number of times the table is used must be counted.
- To set up referencing privileges for dictionary tables, the value obtained by multiplying the number of dictionary tables to be accessed by 5 must be added to the concurrently accessible base tables count (for details about the dictionary table referencing privilege, see the *HiRDB Version 8 System Operation Guide*).

7) pd_utl_exec_mode = 0 | 1

Specifies a value that indicates the maximum number of concurrent executions of utilities, as indicated below:

Utility	Maximum number of concurrent executions	
	pd_utl_exec_mode = 0	**pd_utl_exec_mode = 1**
Database load utility	32	`pd_max_users value`
Database reorganization utility	32	
Database copy utility*	32	
Database recovery utility	32	
Database condition analysis utility	16	
Optimization information collection utility	16	`pd_max_users value ÷ 2`

* The number of backup files affects the maximum number of concurrent executions of the database copy utility. As the number of backup files increases, the maximum number of concurrent executions decreases. For details, see the explanation of the database copy utility in the *HiRDB Version 8 Command Reference.*

Specification guidelines

- Specifying `0` fixes the size of the HiRDB shared memory to be used by utilities.
- When `1` is specified, the shared memory needed for utility execution is allocated dynamically during utility execution.
- When the value of `pd_max_users` is 32 or less and the number of concurrent executions of utilities is less than the value specified in `pd_max_users`, specifying `pd_utl_exec_mode = 1` will reduce the amount of shared memory that is used.
- If a HiRDB/Parallel Server satisfies the following condition, specify `1` for this operand.

  total-number-of-BESs × 2 × *number-of-utilities-to-be-concurrently-executed* ≥ 824

Note

- If the number of back-end servers is large, a shared memory shortage may prevent utility execution.

8) pd_max_commit_write_reclaim_no = ***maximum-number-of-concurrent-executions-of-pdreclaim-commands-with-p-option-specified***

~ <unsigned integer>((0-2000)) <<10>>

Specifies the maximum number of `pdreclaim` commands with the `-p` option specified that can be executed concurrently. For a HiRDB/Parallel Server, this operand specifies the maximum number of concurrent executions per server.

Specifying 0 results in termination with an error of any `pdreclaim` command in which the `-p` option is specified. When the number of concurrent executions of `pdreclaim` commands in which the `-p` option is specified exceeds the value of this operand, the excess `pdreclaim` commands in which the `-p` option is specified terminate with an error.

Specification guidelines

If you specify the `-p` option in `pdreclaim` commands, specify the number of concurrent executions per server that executes those commands. If you always omit the `-p` option from the `pdreclaim` command, specify 0 (to reduce the size of shared memory used).

Notes

If you specify a large value for this operand, check and, if necessary, revise the memory requirement, because the shared memory used by the back-end server, dictionary server, and single server increases. For details about estimating the shared memory, see the *HiRDB Version 8 Installation and Design Guide*.

Relationship to other operands

If `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand, 0 is assumed when this operand is omitted. This means that if you specify `v6compatible` or `v7compatible` in the `pd_sysdef_default_option` operand and omit this operand, `pdreclaim` commands in which the `-p` option is specified cannot be executed.

2.3.3 Operands related to HiRDB startup

9) pd_mode_conf = AUTO | MANUAL1 | MANUAL2

Specifies the HiRDB startup method.

`AUTO`:

HiRDB is automatically started. The automatic startup method automatically starts HiRDB when the OS is started.

However, you must manually start HiRDB in the following cases. In the manual startup method, you must execute the `pdstart` command to start HiRDB.

- Normal start following normal termination (without restarting the OS)
- Restart following planned termination (without restarting the OS)
- Restart following forced termination

MANUAL1:

HiRDB is manually started. However, after abnormal termination, HiRDB is automatically started.

MANUAL2:

Manual startup.

Specification guidelines

- To restart HiRDB (or the unit) automatically after HiRDB (or the unit) has terminated abnormally, specify AUTO or MANUAL1.
- If you use the system switchover facility, specify MANUAL1 or MANUAL2. Which of these modes is more appropriate depends on how the system switchover facility is used. For the methods of operating the system switchover facility, see the *HiRDB Version 8 System Operation Guide*.
- The startup mode is determined by the combination of the previous termination mode (normal termination, forced termination, planned termination, or abnormal termination) and the startup method specified by this operand, as shown in the following table:

pd_mode_conf specification	Previous termination mode	Startup mode	Startup method
AUTO	Normal termination	Normal startup	Automatic startup or Manual startup[1]
	Planned termination	Restart[2]	
	Forced termination	Restart[2]	Manual startup
	Abnormal termination	Restart	Automatic startup
MANUAL1	Normal termination	Normal startup	Manual startup
	Planned termination	Restart[2]	
	Forced termination	Restart[2]	
	Abnormal termination	Restart	Automatic startup

pd_mode_conf specification	Previous termination mode	Startup mode	Startup method
MANUAL2	Normal termination	Normal startup	Manual startup
	Planned termination	Restart[2]	
	Forced termination	Restart[2]	
	Abnormal termination	Restart[2]	

[1] Automatic startup is used only when the OS is started. For normal start following normal termination and restart following planned termination (without restarting the OS), manual startup is used.

[2] The dbdestroy option of the pdstart command can be used for a forced startup. However, when forced startup is used, the HiRDB database is not recovered automatically and must be recovered by the HiRDB administrator.

- If you use Real Time SAN Replication based on the log-only synchronous method, the startup mode is always set to normal startup regardless of the value of this operand or the previous termination mode. In the case of a log application site, the changeable system definition operands are only those that can be changed during restart.

Note

- To use automatic startup (pd_mode_conf=AUTO) in a HiRDB/Parallel Server, try to start all units within 20 minutes after starting the first unit. If all units are not started within 20 minutes, the HiRDB startup process is cancelled. You can use the pd_reduced_check_time operand to change this 20-minute time limit.

  Note that this time limit does not apply when you are restarting a unit after it or the OS was abnormally terminated.
- If you use Real Time SAN Replication, AUTO cannot be specified. If AUTO is specified, HiRDB startup processing results in an error.
- If you use Real Time SAN Replication based on the log-only synchronous method, MANUAL2 is assumed even when MANUAL1 is specified in this operand at the log application site.

10) pd_system_complete_wait_time = *pdstart-command-completion-wait-time*

~ <unsigned integer>((610-3600))<<610>>(Seconds)

Specifies the amount of time to wait for completion of the pdstart command's processing. This operand needs to be specified only when it is necessary to provide a longer wait time than the default value of 610 seconds (10 minutes and 10 seconds). If startup processing is not completed within the specified number of seconds after the pdstart command was entered, HiRDB outputs the KFPS05078-I message and returns the pdstart command with an error.

Even though the pdstart command is returned with an error, startup processing continues and HiRDB is started. Therefore, this operand should be specified only in the following cases:

- When the KFPS05078-I message is the monitoring target
- When another operation is to be performed automatically after the pdstart command terminates normally

Application criterion

Normally, this operand need not be specified. This operand is applied when the KFPS05078-I message is output and the pdstart command is returned with an error during HiRDB startup processing.

11) pd_start_time_out = ***HiRDB*****-start-preparation-maximum-wait-time**

~ <unsigned integer>((1-1440))<<15>>(minutes)

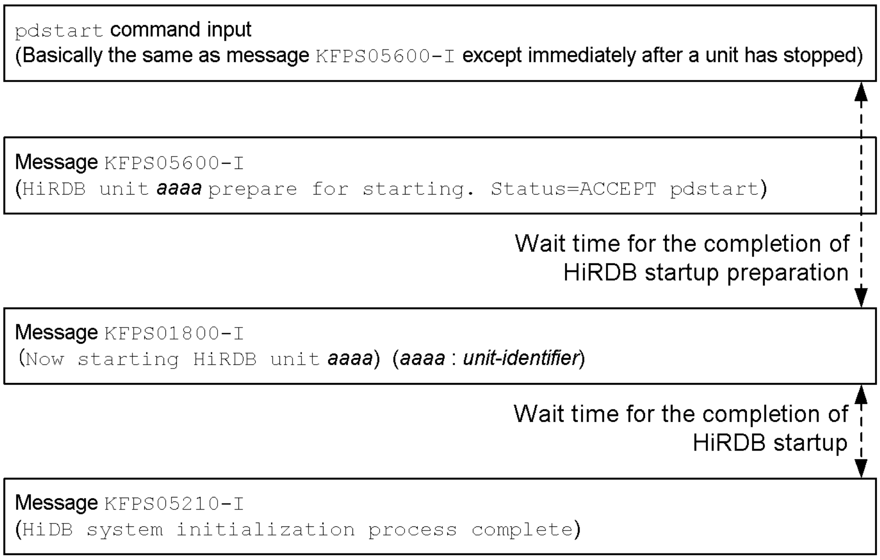
Specifies the maximum amount of time to wait for completion of HiRDB start preparation after pdstart command input. If HiRDB start preparation is not completed within the time specified in this operand after pdstart command input, the KFPS01861-E message (reason code = TIMEOUT) is output in the pdstart command input window.

Specification guidelines

Normally, you need not specify this operand. In an ordinary environment, the HiRDB start preparation time is at most 3 to 6 minutes, and therefore the default value [15 minutes] of this operand is sufficient for the maximum wait time. In rare cases, HiRDB start preparation may take more than 15 minutes because of factors such as server machine performance, disk size, and memory load. In this case, the KFPS01861-E message (reason code = TIMEOUT) is output in the pdstart command input window. If this occurs, specify a value greater than 15 minutes for this operand.

Reference note:

Based on the time at which the following message was output, you can compute the wait time for the completion of the HiRDB startup preparation process and the wait time for the completion of the HiRDB startup:



pdstart command input
(Basically the same as message KFPS05600-I except immediately after a unit has stopped)

Message KFPS05600-I
(HiRDB unit *aaaa* prepare for starting. Status=ACCEPT pdstart)

Message KFPS01800-I
(Now starting HiRDB unit *aaaa*) (*aaaa* : *unit-identifier*)

Message KFPS05210-I
(HiDB system initialization process complete)

Wait time for the completion of
HiRDB startup preparation
Wait time for the completion of
HiRDB startup

12) pd_term_watch_count =
maximum-number-of-successive-abnormal-terminations

~ <unsigned integer>((1-3))<<3>>

Specifies the maximum number of times in a row that abnormal termination can occur. If HiRDB terminates abnormally during HiRDB (or unit) restart attempts for the number of times in a row specified in this operand, restart processing is canceled. For example, if `2` is specified here, restart processing is canceled after two HiRDB restarts in a row terminate abnormally.

The HiRDB restart attempts can be either automatic or manual, depending on the value specified in the `pd_mode_conf` operand.

When HiRDB terminates abnormally after the number of abnormal terminations specified in this operand has been reached, the `KFPS00715-E` message is output. In this case, remove the cause of the error and then execute the `pdrpause` command in the unit indicated in the `KFPS00715-E` message.

2.3.4 Operands related to reduced activation

13) pd_start_level = <u>0</u> | 1

This operand is applicable to a HiRDB/Parallel Server.

Specifies whether or not reduced activation is to be used when a unit cannot be started during HiRDB startup. For details about reduced activation, see the *HiRDB Version 8 System Operation Guide.*

`0`:

Do not use reduced activation. HiRDB will not start if an error prevents any unit from being activated.

`1`:

Use reduced activation. Even if an error prevents a unit from being activated, HiRDB (all other units) will be started.

14) pd_reduced_check_time = *wait-time-for-reduced-activation-startup-notice*

～ <unsigned integer>((300-1200))<<1200>>(seconds)

This operand is applicable to a HiRDB/Parallel Server.

Specifies the amount of time to wait for receipt of a notice of activation completion from each unit. If the activation completion notice is not received from a unit before the amount of time specified here elapses, reduced activation without that unit is used.

Condition

`1` must be specified for the `pd_start_level` operand.

Specification guidelines

Before specifying this operand, you should determine how long it takes for the `KFPS01826-I` message to be output after the `pdstart` command is entered (the `KFPS01826-I` message is output to the unit that defines the system manager). The value specified here must be greater than this message's output time.

Notes

Even when the wait time specified by this operand elapses, reduced activation cannot be executed if all the conditions for reduced activation have not been satisfied. In this case, HiRDB is abnormally terminated. For details about the conditions for reduced activation, see the *HiRDB Version 8 System Operation Guide*.

15) pd_start_skip_unit = *name-of-unit-not-to-be-started*[,*name-of-unit-not-to-be-started*]...

This operand is applicable to a HiRDB/Parallel Server.

Specifies the names of units that need not be started when reduced activation is executed. The units specified here are excluded from the startup process.

Condition

The `pd_start_level operand` must be set to `1`.

Advantage

During a startup, HiRDB waits for up to 20 minutes for activation startup notifications from all units. If reduced activation is in effect, HiRDB will wait the full 20 minutes for units that cannot be started. If units that cannot be started are specified in this operand, HiRDB will not wait for activation startup notifications from them, thus saving up to 20 minutes of wait time and reducing the time for reduced activation to be implemented.

Note

- If the standby-less system switchover (effects distributed) facility is being used, not all units within the HA group need to be active, and therefore specification of this operand is invalid. However, all back-end servers within the HA group must be running at some unit for HiRDB to be able to start.
- Even if a recovery-unnecessary front-end server unit is specified for this operand, that server unit is not treated as a reduced activation target.

2.3.5 Operands related to HiRDB processing

16) pd_dbsync_point = <u>sync</u> | commit

Specifies the timing for committing database updates to the file.

`sync`:

Commit database updates to the file at each synchronization point. This option enhances performance when a large number of transactions that update the same page occur between synchronization points. Because update information is not committed to the file when a `COMMIT` statement is issued, the input/output workload is reduced. Note that full recovery processing is slower than when `commit` is specified.

`commit`:

Commit database updates to the file when a `COMMIT` statement is issued. Because the database contents are guaranteed when the transaction is completed, there is no need to recover transaction processing from a synchronization point, thus reducing the time required for a full recovery. However, if a large number of transactions that update the same page occur between synchronization points, this option is slower than when the `sync` option is specified.

Note

In the following cases, either specify `sync` in this operand or omit this operand:

- Real Time SAN Replication based on the hybrid method is used.
- Real Time SAN Replication based on the log-only synchronous method

is used.

Remarks

A LOB RDAREA is not affected by this operand. Directories are updated when the `COMMIT` statement is issued. Whether or not data is updated depends on whether a LOB global buffer has been allocated. If no LOB global buffer has been allocated, data is instantly updated when an update request is issued. If a LOB global buffer has been allocated, data is updated when the `COMMIT` statement is issued. However, if the global buffer becomes full, data is updated at that time.

Relationship to other operands

This operand is related to the `pd_system_dbsync_point` operand.

17) pd_system_dbsync_point = sync | <u>commit</u>

Specifies the timing for committing to file updates in the following types of RDAREAs:

- Master directory RDAREA
- Data directory RDAREA
- Data dictionary RDAREAs
- Data dictionary LOB RDAREAs
- Registry RDAREAs
- Registry LOB RDAREAs

`sync`:

Commit RDAREA updates to the file at each synchronization point. Because update information is not committed to the file when a `COMMIT` statement is issued, the processing performance of a definition SQL is slightly better than when the `commit` option is specified. However, full recovery processing is slower than when the `commit` option is specified.

`commit`:

Commit to file updates to the indicated types of RDAREAs when a `COMMIT` statement is issued. Because the contents of updates to the indicated types of RDAREAs are guaranteed when the transaction is completed, there is no need to recover these types of RDAREAs from a synchronization point, thus reducing the time required for a full recovery. However, the processing performance of a definition SQL is slightly lower than when `sync` is specified.

Relationship to other operands

This operand is related to the `pd_dbsync_point` operand. The following table shows the relationship to the `pd_dbsync_point` operand:

pd_dbsync_point specification	pd_system_dbsync_point specification	
	sync	commit (default)
`sync` (default)	Commits updates to all RDAREAs at a synchronization point.	Commits updates to the indicated types of RDAREAs when a `COMMIT` statement is issued. Commits updates to other types of RDAREAs at a synchronization point.
`commit`	Commits updates to all RDAREAs when a `COMMIT` statement is issued.	

18) pd_dbsync_altwrite_skip = Y | N

Specifies the handling of the write processing when an update buffer reference request is issued during synchronization point acquisition processing. Normally, such write processing of the update buffer contents into the database is handled by the transaction process that issues the reference request. This operand specifies whether or not the handling of this write processing is to be skipped.

The following compares the values of this operand.

Comparison item	pd_dbsync_altwrite_skip operand value	
	Y	N (default value)
Database writing method when an update buffer reference request is issued during synchronization point acquisition processing	The transaction process that issued the reference request does not handle the writing of the update buffer contents into the database (the handling is skipped).	The transaction process that issued the reference request handles the writing of the update buffer contents into the database (the handling is not skipped).
Advantage	Because the transaction process does not handle the write processing, the performance of the referencing transaction is stable during synchronization point acquisition processing.	The amount of time required for synchronization point acquisition processing is reduced because some of the workload is distributed to the referencing transaction.
Disadvantage	The amount of time required for synchronization point acquisition processing increases because none of the workload is distributed to the referencing transaction.	There may be adverse effects on the referencing transaction during synchronization point acquisition processing.

Specification guidelines

You can execute the statistics analysis utility to obtain statistical information on the global buffer pool in order to check the database write processing take-over count (`ALTRW`) due to reference request hits during synchronization point processing. If this value is large, performance of a referencing transaction is not stable during synchronization point acquisition processing. To achieve stable performance, specify `Y`. Note that when `Y` is specified, more time is required for synchronization point acquisition processing; if necessary, you can use the facility for parallel writes in deferred write processing to distribute the write processing workload.

19) pd_overflow_suppress = Y | N

Specifies whether or not error suppression is to be implemented during computations. The following types of errors can be suppressed:

- Overflow in the middle of a computation
- Division by 0 errors

`Y`:

If an applicable error occurs in a computation during SQL execution, convert the computation result to the null value and continue the processing.

`N`:

If an applicable error occurs in a computation during SQL execution, cancel the process with an error.

20) pd_process_terminator = resident | fixed | nonresident

If a HiRDB process is abnormally terminated, HiRDB starts a process that executes *post-processing*. This operand specifies whether the post-processing process is activated when HiRDB is started.

`resident`:

This option starts a single post-processing process when starting HiRDB. For a HiRDB/Parallel Server, a post-processing process is started in each unit.

If multiple processes are abnormally terminated at the same time, post-processing processes up to the number specified by HiRDB are started and executed in parallel. If a new post-processing process cannot be started due to memory shortage, for example, post-processing is sequentially performed using the post-processing processes that are already active.

`fixed`:

When starting HiRDB, this option starts the number of post-processing processes specified by the `pd_process_terminator_max` operand. For a HiRDB/Parallel Server, the number of post-processing processes specified by the `pd_process_terminator_max` operand are started in each unit. If post-processing processes cannot be started due to memory shortage, for example, HiRDB (or the applicable unit for a HiRDB/Parallel Server) is not started.

If a number of processes exceeding the value specified in the pd_process_terminator_max operand are abnormally terminated at the same time, no additional post-processing processes are started. In this case, post-processing is sequentially performed using the post-processing processes that are already active.

nonresident:

This option does not start any post-processing process when starting HiRDB. A post-processing process is started whenever a process is abnormally terminated.

If multiple processes are abnormally terminated at the same time, post-processing processes are simultaneously started and executed in parallel. If post-processing processes cannot be started due to memory shortage, for example, HiRDB (or the applicable unit for a HiRDB/Parallel Server) may be abnormally terminated in some cases.

Specification guidelines

- To improve reliability, specify resident or fixed. Although fixed provides higher post-processing performance than resident, fixed requires more memory.
- When nonresident is specified, post-processing processes are started on demand. Consequently, post-processing processes cannot be started if memory shortage occurs. Furthermore, if multiple processes are abnormally terminated at the same time, multiple post-processing processes are started, resulting in performance degradation.

Note

You must be careful when changing the specification value to fixed. Because this option starts post-processing processes when starting HiRDB, it requires more memory. If memory shortage, for example, prevents post-processing processes from being started, HiRDB (or the applicable unit for a HiRDB/Parallel Server) cannot start.

21) pd_process_terminator_max = ***maximum-number-of-resident-post-processing-processes***

~ **<unsigned-integer>((1-100)) << (3, ↑**(value of pd_max_users + value of pd_max_reflect_process_count)/100**↑**)>>

Specify this operand if you have omitted the pd_process_terminator operand or specified fixed or it. Specify for the pd_process_terminator_max operand the number of post-processing processes to be started when starting HiRDB. If memory shortage, for example, prevents the specified number of post-processing processes from being started, HiRDB (or the applicable unit for a HiRDB/Parallel Server) cannot start.

Specification guidelines

The number of post-processing processes needed is proportional to the value of `pd_max_users` + `pd_max_reflect_process_count`. Take this into consideration when determining a value for the `pd_process_terminator_max` operand.

Relationship to other operands

If `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand, the default value for this operand is as follows:

↑(value of `pd_max_users` + value of `pd_max_reflect_process_count`)/100↑

22) pd_space_level = 0 | 1 | 3

Specifies the applicable space conversion level when the space configuration facility is used. For details about the space conversion facility, see the *HiRDB Version 8 System Operation Guide.*

`0`: Use space conversion level 0.

`1`: Use space conversion level 1.

`3`: Use space conversion level 3.

The following table explains the space conversion levels.

Level	Explanation
Level 0	No space conversion.
Level 1	Converts spaces in literals, embedded variables, and ? parameter data in a data manipulation SQL as follows: • If a character string literal is considered a national character string literal, two single-byte spaces are converted into a double-byte space. If a single-byte space appears by itself, it is not converted. • If a character string literal is considered a mixed character string literal, a double-byte space is converted into two single-byte spaces. • When data is being stored into a national character string-type string or is being compared with a national character string-type value expression, two single-byte spaces in an embedded variable or ? parameter are converted into a double-byte space. If a single-byte space appears by itself, it is not converted. • When data is being stored into a mixed character string-type string or is being compared with a mixed character string-type value expression, a double-byte space in an embedded variable or ? parameter is converted into two single-byte spaces.

Level	Explanation
Level 3	The following processing is performed, in addition to the processing in Level 1: • When data of a national character string-type value expression is being retrieved, a double-byte space is converted into two single-byte spaces.

Notes

- When the space conversion level is changed, there may be differences in UAP results before and after the change. To ensure that UAP results remain the same, the space conversion level should not be changed.
- When sorting is performed with the space conversion level set to 3, HiRDB performs space conversion on the sorting results, which may produce unexpected results.
- When data is being stored in a cluster key string, space conversion may cause a uniqueness error. In such a case, either store the data without space conversion or use the database reorganization utility to standardize the spaces in the existing database.
- Spaces in a national character string are converted in units of two characters from the beginning of the string.
- The following should be noted when 1 or 3 is specified for the space conversion level:

  When a storage RDAREA is determined in the case of a UAP that uses a hash function for table partitioning from a hash-partitioned table, it is necessary to specify a space conversion level in the argument of the hash function for table partitioning. If no level is specified, the result of the hash function for table partitioning may be incorrect.

  When a UAP is used to execute key range partitioning on a table that has been key range-partitioned (partitioning key is national character data or mixed character data), the partitioning key must be converted with the space conversion function. Otherwise, the result of key range partitioning may be incorrect.

  For details about the hash functions for table partitioning and the space conversion function, see the *HiRDB Version 8 UAP Development Guide.*

Relationship to client environment definition

The value of this operand can be different for each client. To change the operand for a client, specify the `PDSPACELVL` operand in the client environment definition.

For details about the `PDSPACELVL` operand, see the *HiRDB Version 8 UAP*

Development Guide.

23) pd_dec_sign_normalize = Y | N

Specifies whether or not the sign of signed packed data transferred from a UAP is to be normalized (i.e., specifies whether or not the facility for conversion to a DECIMAL signed normalized number is to be used).Normalizing the sign means converting the sign A-F of signed packed data to C or D. For details about the facility for conversion to a DECIMAL signed normalized number, see the *HiRDB Version 8 System Operation Guide.*

`Y`:

Use the facility for conversion to a DECIMAL signed normalized number. The sign of signed packed data is normalized. A-F are considered to be DECIMAL data signs.

`N`:

Do not use the facility for conversion to a DECIMAL signed normalized number. The sign of signed packed data is not normalized. C, D, and F are considered to be DECIMAL data signs.

Notes

The following must be noted when Y is specified (i.e., when the facility for conversion to a DECIMAL signed normalized number is to be used):

- For determining the storage RDAREA for a hash-partitioned table in the case of a UAP that uses the hash function for table partitioning, it is necessary to specify "Use the facility for conversion to a DECIMAL signed normalized number" as an argument of the hash function for table partitioning. If this is not specified, the result of the hash function for table partitioning may be corrupted.
- When a UAP is used to execute key range partitioning for a key range-partitioned table (for data whose partitioning key is DECIMAL), the partitioning key value must be converted using the facility for conversion to a DECIMAL signed normalized number. Otherwise, the result of the key range partitioning may be corrupted.

For details about the hash functions for table partitioning and the facility for conversion to a DECIMAL signed normalized number, see the *HiRDB Version 8 UAP Development Guide.*

24) pd_server_entry_queue = spnfifo | fifo | loop

If contention occurs in the HiRDB server process during concurrent execution of UAPs, processing requests may sometimes be temporarily queued. This operand specifies what HiRDB must do in this case. Note that *process contention* in this case means that multiple processes are simultaneously trying to lock internal resources, such as tables and RDAREAs, when transactions are running on the HiRDB server process. Only a single process is allowed to lock internal resources at any point in time. *Spin*, referred to in the following explanation means a process for acquiring the right to execute a lock. When another process releases the right to execute a lock, a process that is spinning has a higher probability of acquiring the right to execute a lock.

`spnfifo`:

A processing request that occurs first is given higher priority. However, because the process is spun a certain number of times before being registered in a queue, the priority order is not perfect. This method is used in Version 06-00 and older versions.

`fifo`:

A processing request that occurs first is given higher priority than when `spnfifo` is specified. Because no spinning occurs before a process is registered in a queue, the priority order is maintained better than when `spnfifo` is specified. This method also reduces the CPU load.

`loop`:

All processing requests are given the same priority. When processes are registered in a queue, they are spun at high speed. Specifying `loop` may improve the response during concurrent execution of UAPs. However, this method places a greater load on the CPU than other methods.

Specification guidelines

Normally, you need not specify this operand.

Change the specification value if the processing performance during concurrent execution of UAPs does not improve. Doing so may improve the performance.

25) pd_thdlock_sleep_func = <u>0</u> | 1

Specifies the type of sleep function (for waiting for a specified amount of time) to be used when acquiring a lock on shared resources, such as shared memory. The following table shows the relationship between the value specified by this operand and the characteristics of the function issued by HiRDB.

pd_thdlock_sleep_func operand value	Characteristics of the function issued by HiRDB	
	Process switchover rate	CPU usage
`0`	Low	Low
`1`	High	High

Specification guidelines

- Normally, you need not specify this operand.
- Change the specification value of this operand if the concurrent execution performance of UAPs is significantly worse than stand-alone performance. Doing so may improve the performance.
- Whether `0` or `1` produces better performance depends on the OS type, machine performance, disk performance, UAP details, and the number of UAPs being concurrently executed. However, specifying `0` usually produces relatively stable performance.

Notes

If you change the specification value of this operand from `0` to `1`, the CPU usage may increase, resulting in performance degradation in some cases.

Relationship to other operands

Specification of this operand is invalid if a value greater than `10,000` is specified for the `pd_thdlock_retry_time` operand.

26) pd_thdlock_wakeup_lock = Y | N

Specifies a thread lock release notification method.

`Y`:

When issuing a thread lock release notification, a new separate lock is temporarily obtained. Because this temporary lock is obtained, reliable transmission of the release notification is guaranteed.

`N`:

When issuing a thread lock release notification, no new separate lock is temporarily obtained. When no temporary lock is obtained, a transaction may occur that has a longer execution time than other transactions. How much longer is determined by the value specified for the `pd_thdlock_pipe_retry_interval` operand.

Specification guideline

1. The following table shows the effects of multiplexed transaction execution:

Comparison item	pd_thdlock_wakeup_lock operand value	
	Y	N
Difference in transaction execution time	Reduces the difference.	Increases the difference.
Time required for the completion of all transactions	Lengthens	Shortens

2. The following table shows the recommended values of this operand:

Condition		Recommended value
When a new HiRDB is used		Y
When HiRDB is already being used	When the execution times of all transactions must be the same during multiplexed transaction execution	Y
	When some differences in the execution times of all transactions are allowed during multiplexed transaction execution	N

Relationship to other operands

If you specify N in this operand or you omit this operand, evaluate the specification of the pd_thdlock_pipe_retry_interval operand.

27) pd_thdlock_pipe_retry_interval = *thread-lock-release-check-interval*

~ <unsigned integer>((0-2147483647))<<100000>>(microseconds)

Specifies the interval in microseconds at which to check for thread lock release. Specify this operand if pd_thdlock_wakeup_lock = N (default value) is specified. In all other cases, there is no need to specify this operand.

If multiple transactions are executed concurrently, differences occur in their execution times. The value specified for this operand provides an estimate for the maximum execution time difference (this value is not exact).

Specification guidelines

- If this operand is to be specified, specify 1000 for it.
- If the specification of 1000 increases the CPU usage rate too much, specify a greater value.
- Even if you specify a value less than 1000, performance does not improve over a case in which 1000 is specified.

Note

If 0 is specified for this operand, release check is repeated within short intervals and may result in an extremely high CPU usage rate.

28) pd_thdlock_retry_time = *thread-lock-sleep-time*

~ <unsigned integer>((1-1000000))<<10000>>(microseconds)

Specifies the thread lock sleep time in microseconds. If this operand is specified when all the conditions listed below are satisfied, the CPU usage rate may decrease. Otherwise, there is no need to specify this operand.

- The CPU usage rate is very high.

- Reducing the CPU usage rate is necessary even if it results in a reduction in performance.
- 0 is specified for the pd_thdlock_sleep_func operand.

The following describes the HiRDB processing based on the combination of the pd_thdlock_sleep_func and the pd_thdlock_retry_time operand values:

pd_thdlock_sleep_func operand value	pd_thdlock_retry_time operand value	
	1 to 10000	10001 to 1000000
0	Each process stands by for the thread lock sleep time specified by select() or Sleep().	
1	The OS determines process allocation using sched_yield() or SwitchToThread() (the pd_thdlock_retry_time operand value is ignored).	Each process stands by for the thread lock sleep time specified by select() or Sleep().

Specification guidelines

- If this operand is to be specified, begin specify 10000 for it.
- If the specification of 10000 increases the CPU usage rate too much, specify a greater value.
- If there is an ample margin in the CPU usage rate, specify a value less than 10000.

Notes

- Reducing the value may not change the performance.
- Specifying 1000 or a greater value may have an adverse effect on performance.

Relationship to other operands

If v6compatible is v7compatible is specified in the pd_sysdef_default_option operand, the default value for this operand is 10000.

29) pd_thdspnlk_spn_count = *thread-spin-lock-spin-count*

~ <unsigned integer>((0-2147483647))<<512>>

Specifies a spin count for thread spin lock. Specifying this operand when all of the following conditions are satisfied may improve the system performance. Otherwise, there is no need to specify this operand.

- An ample margin exists in the CPU usage rate.
- You want to improve performance, even if the CPU usage rate must be increased.

Specification guidelines

- If this operand is to be specified, specify a value that is greater than the default value (`512`).
- Because the specification value depends on the OS type, the processor type, the machine performance, the disk performance, the UAP content, and the number of UAPs concurrently being executed, there is no clear guideline. Determine an appropriate value by varying the specification value and measuring the performance.

Notes

- If the value of this operand is too large, the CPU usage rate may increase, causing problems such as slower OS operation. In this case, decrease the operand value.
- Increasing the value of this operand may not always improve performance.

30) pd_pageaccess_mode = SNAPSHOT | NORMAL

Specifies the page access mode to be used for database search.

`SNAPSHOT`:

Uses a snapshot mode for page access. When the global buffer is accessed for the first time, rows that match the search condition are copied to the process private memory. During the second search request, a search result is returned by referencing the process private memory. For details on the snapshot mode, see the *HiRDB Version 8 Installation and Design Guide*.

`NORMAL`:

Uses a normal mode for page access. The global buffer is accessed for each search request.

Specification guidelines

If facilities for improving performance, such as the rapid grouping facility, cannot be used, consider using the snapshot mode. In normal search-SQL, the global buffer is accessed roughly the same number of times as the number of rows that match the specified search condition. Consequently, if search-SQLs are concurrently executed, accesses to the global buffer become concentrated, and as a result, the expected performance may not be obtained. In this case, using the snapshot mode can reduce the number of accesses by search-SQLs to the global buffer, and thus may improve the performance. However, using the snapshot mode increases the size of the process private memory used by HiRDB. For the method of computing the size of process private memory when using the snapshot mode, see the *HiRDB Version 8 Installation and Design Guide*.

31) pd_cmdhold_precheck = Y | N

Specifies whether to check for RDAREA hold before locking the RDAREA.

`Y`:

Checks for RDAREA hold before locking the RDAREA.

`N`:

Does not check for RDAREA hold before locking the RDAREA. Checking is performed after locking.

The following hold types are checked:

- Command hold
- Reference-possible hold
- Reference-possible backup hold

Specification guidelines

Normally, omit this operand or specify `Y`. The following table describes the differences between `Y` and `N` specifications.

Item	Y specified	N specified
Processing by HiRDB	During UAP or command* execution, HiRDB checks the hold statuses of all RDAREAs that store the access target tables before locking the RDAREAs. For example, when accessing a table that is row-partitioned into RDAREAs 1 through 4, HiRDB checks the hold statuses of all these RDAREAs.	During UAP or command* execution, HiRDB first locks RDAREAs and then checks the hold statuses of the access target RDAREA. For example, suppose that a table that is row-partitioned into RDAREAs 1 through 4 is to be accessed. If the target RDAREAs are narrowed using an index and if RDAREA 1 is to be accessed, HiRDB checks the hold status of RDAREA 1 only. This mode is used in HiRDB Version 5.0 and older versions.
When a UAP accesses an RDAREA that is on hold	Because a hold check is performed before locking the RDAREA, the fact that the RDAREA is on hold can be detected more quickly than when `N` is specified.	Because a hold check is performed after locking the RDAREA, the locked RDAREA may cause a timeout error (`KFPA11770-E`) if a UAP accesses the RDAREA that is on hold. Additionally, if the access target RDAREA is on hold because data is being loaded or because it is being reorganized, the UAP may cause a hold error (`KFPA11920-E`).

Item	Y specified	N specified
When using a non-row partitioning index to narrow the access target RDAREAs	You must be careful when a table is row-partitioned but the index is not. When using a non-row partitioning index to narrow the access target RDAREAs. a hold error (KFPA11920-E) occurs, even when a non-access target RDAREA is on hold. In the example given for processing by HiRDB, the UAP causes a hold error (KFPA11920-E) if any of RDAREAs 1 through 4 is on hold.	When using a non-row partitioning index to narrow the access target RDAREAs, the UAP or command can be executed even if a non-access target RDAREA is on hold. In the example given for processing by HiRDB, the UAP can be executed even if RDAREAs 2 through 4 are on hold.

* Refers to UAPs and commands that cannot be executed if RDAREAs are on hold.

32) pd_db_io_error_action = dbhold | unitdown

Specifies the processing to be performed by HiRDB when an input/output error occurs in an RDAREA (excluding the master directory RDAREA). If an input/output error occurs in the master directory RDAREA, HiRDB (or a unit for a HiRDB/Parallel Server) is always abnormally terminated regardless of the specification in this operand. For the actions to be taken when an RDAREA input/output error occurs, see the *HiRDB Version 8 System Operation Guide*.

An input/output error in this case refers to an error that occurs when a file manipulation attempt by HiRDB fails due to a cause that cannot be determined by HiRDB. When such an error occurs, `-1544` is output as the error code returned in response to a HiRDB file system access request.

`dbhold`:

When an input/output error occurs in an RDAREA, the RDAREA is placed in an error shutdown state.

`unitdown`:

If an input/output error occurs in an RDAREA, HiRDB (or a unit for a HiRDB/Parallel Server) is abnormally terminated. However, if an input/output error occurs again following abnormal termination, the RDAREA is placed in an error shutdown state. To enable the specification of `unitdown` again, take one of the following actions:

- Start HiRDB normally.
- Execute the system reconfiguration command (`pdchgconf` command).

Specification guidelines

To determine the specification value for this operand, see *Actions to be taken* *when an RDAREA input/output error occurs* in the *HiRDB Version 8 System Operation Guide*.

Notes

- When an input/output error occurs when `unitdown` is specified, HiRDB is abnormally terminated. Consequently, if a UAP or utility is being executed in the pre-update log acquisition mode or no-log mode, the processing target RDAREA may go into an error shutdown state.
- If an input/output error occurs during the startup or termination process, HiRDB is not abnormally terminated even if `unitdown` is specified.
- If Real Time SAN Replication based on the log-only synchronous method is used and an input/output error occurs during log application processing at the log application site, HiRDB does not terminate abnormally even when `unitdown` is specified.

33) pd_connect_errmsg_hide = Y | N

Specifies whether to hide the error cause in the message that is output when a connection attempt fails.

`Y`: Hides the error cause when a connection attempt fails.

`N`: Does not hide the error cause when a connection attempt fails.

Depending on the value specified for this operand, the message that is output when a connection attempt fails may vary. The following table shows the details:

Error cause	Output message	
	pd_connect_errmsg _hide = Y	pd_connect_errmsg _hide = N (default value)
Invalid authorization identifier (the specified user does not exist)	KFPA19632-E	KFPA11561-E
Invalid password (the specified password is invalid)	KFPA19632-E	KFPA11560-E

34) pd_cancel_down_msgchange = Y | N

Specifies whether or not the error messages output when a server process is terminated forcibly are to be changed.

`Y`:

Changes the error messages to warning messages. The facility for changing error messages is called the facility for changing a process down message when cancelling a transaction.

N:

Does not change the error messages.

The following shows the relationship between the value of this operand and the error messages that are output:

Condition	Messages that are output	
	When Y (default value) is specified	When N is specified
Server process is terminated forcibly for one of the following reasons:# • Intentional forced termination by the user • Forced termination caused by a timeout • Forced termination due to a failure at the client	• KFPS01852-W • KFPO00115-W	• KFPS01820-E • KFPO00105-E
Server process is terminated forcibly for some other reason	• KFPS01820-E • KFPO00105-E	
HiRDB cannot identify the cause of the forced termination of the server process		

#

There are other causes that change the messages. For details, see *Facility for changing a process down message when cancelling a transaction* in the *HiRDB Version 8 System Operation Guide*.

Advantages

By specifying Y in this operand, you change the messages that are displayed for identifying the cause of forced termination of a server process.

Remarks

When the KFPS01820-E and KFPO00105-E messages are displayed, it is not possible to use the message IDs to distinguish between errors detected by HiRDB and errors resulting from an intentional user operation. To identify the cause, you must compare the process IDs that are displayed in the individual messages.

If JP1 is used to monitor messages, handling based on the KFPS01820-E and KFPO00105-E messages may be complicated because information about multiple messages cannot be compared. Specifying Y in this operand makes it easier to handle such messages because the output messages are classified by error cause. For this reason, it is recommended that you specify Y in this operand when you use JP1 to monitor messages.

Relationship to other operands

If v6compatible or v7compatible is specified in the pd_sysdef_default_option operand, the default value for this operand is N.

2.3.6 Operands related to full recovery processing

35) pd_max_recover_process = ***concurrently-executable-full-recovery-processes-count***

~ <unsigned integer>((1-10))<<3>>

Specifies the number of processes to be recovered during full recovery processing. For a HiRDB/Parallel Server, this operand specifies the number of processes to be recovered per server (dictionary server or back-end server).

Condition

Parallel input/output processing is not performed for regular files.

Specification guidelines

- The value to be specified depends on the file type:

Condition		Specification guideline
Character special files	There are at least 10 character special files (per server) that define RDAREAs	10
	There are fewer than 10 character special files (per server) that define RDAREAs	Number of character special files
Regular file		1

- Increasing the value of this operand increases the input/output concurrency during full recovery processing, and thus can shorten the recovery time. However, because a number of processes equaling *value-of-this-operand* × *server-count* are started, determine a value by taking the aforementioned specification value guideline and HiRDB resources into consideration.

36) pd_redo_allpage_put = Y | N

Specifies whether to output to a database the pages that were updated after a synchronization point during full recovery processing.

Y:

During full recovery processing, all pages updated after a synchronization point are output.

N:

During full recovery processing, only those pages that were not output to the database when an error occurred are output.

Specification guidelines

Normally, there is no need to specify this operand.

When a database is mirrored using LVM (Logical Volume Manager) or the device driver mirroring facility, the original and duplicate of the mirror may not match each other if an OS or machine error occurs or if a unit error results in system switchover. In this case, to synchronize the original and duplicate volumes:

1. First use LVM or the device driver facility to synchronize the original and duplicate volumes, and then restart HiRDB.
2. If you cannot use the above method to synchronize the original and duplicate volumes, or if the synchronization operation takes too long to satisfy the system requirements such as the system switchover time, specify `Y` for this operand. During full recovery processing at the restart of HiRDB following an error, all pages that were updated after a synchronization point are output to a database. In this process, HiRDB synchronizes the original and duplicate volumes, eliminating the mismatch between the original and duplicate mirrors.

For the action to take when a mismatch occurs between the original and duplicate mirrors, see the *HiRDB Version 8 System Operation Guide*.

Notes

- When `Y` is specified for this operand, there is overhead for outputting all pages to the database during full recovery processing.
- When this operand is specified, the message `KFPH24004-I` is output when the full recovery processing is finished. No message is output if this operand is omitted.

2.3.7 Operands related to transaction decision processing

37) pd_trn_rerun_branch_auto_decide = Y | N

This operand is applicable to a HiRDB/Parallel Server.

Specifies whether or not an undecided transaction that has branched from a transaction is to be decided automatically when a unit is restarted. An undecided transaction occurs when a unit terminates abnormally while the latter transaction's first phase of `COMMIT` is incomplete.

`Y`:

Decide undecided transactions automatically.

`N`:

Do not decide undecided transactions automatically. In this case, the HiRDB administrator must make the decision for an undecided transaction. For details about how to decide an undecided transaction, see the *HiRDB Version 8 System Operation Guide.*

Notes

The following should be noted when `Y` is specified or when specification of this operand is omitted:

- If the reduced activation facility is used, transactions on a unit that is not running at the time of reduced activation are not subject to automatic decision. Therefore, when you restart a unit that was not running during reduced activation, such as after error recovery, check for any undecided transactions. Any undecided transactions that exist must be decided. For details about how to check for undecided transactions and how to decide them, see the *HiRDB Version 8 System Operation Guide*.
- The volume of system log information at the server where the branching-source transaction resides will increase.
- While an undecided transaction is being decided, write-protection is applied to the system log at the server where the branching-source transaction resides.

38) pd_trn_send_decision_intval_sec = *transmission-retry-interval-in-seconds-for-automatic-transaction-decision*

~ <unsigned integer>((0-65535)) <<15>> (seconds)

This operand is applicable only to HiRDB/Parallel Server.

Specifies the interval (in seconds) for sending an automatic decision instruction to a branched transaction when the previous send operation has failed for some reason.

Condition

`Y` must be specified in the `pd_trn_rerun_branch_auto_decide` operand, or the `pd_trn_rerun_branch_auto_decide` operand must be omitted.

Notes

Specifying `0` increases the communication workload, because the decision instruction is resent continuously.

Relationships to other operands

This operand has the following relationships with the `pd_trn_send_decision_interval` operand:

- To specify the time before re-transmission in seconds, use the pd_trn_send_decision_intval_sec operand; to specify it in minutes, use the pd_trn_send_decision_interval operand.
- If the pd_trn_send_decision_intval_sec and the pd_trn_send_decision_interval operands are both specified, the pd_trn_send_decision_intval_sec operand takes precedence.
- If the pd_trn_send_decision_intval_sec and the pd_trn_send_decision_interval operands are both omitted, the following value is assumed depending on the pd_sysdef_default_option operand value:

pd_sysdef_default_option operand value	Default value
recommendable	Default value of the pd_trn_send_decision_intval_sec operand (15 seconds)
v6compatible or v7compatible	Default value of the pd_trn_send_decision_interval operand (5 minutes)

39) pd_trn_send_decision_interval = *transmission-retry-interval-in-minutes-for-automatic-transaction-decision*

~ <unsigned integer>((0-65535)) (minutes)

This operand is applicable only to HiRDB/Parallel Server.

Specifies the interval (in minutes) for sending an automatic decision instruction to a branched transaction when the previous send operation has failed for some reason. Normally, you will specify the pd_trn_send_decision_intval_sec operand and omit this operand.

Condition

Y must be specified in the pd_trn_rerun_branch_auto_decide operand, or the pd_trn_rerun_branch_auto_decide operand must be omitted.

Notes

Specifying 0 increases the communication workload, because the decision instruction is resent continuously.

Relationships to other operands

If v6compatible or v7compatible is specified in the pd_sysdef_default_option operand, the default value for this operand is 5.

This operand has the following relationships with the pd_trn_send_decision_interval operand:

- To specify the time before re-transmission in seconds, use the pd_trn_send_decision_intval_sec operand; to specify it in minutes, use the pd_trn_send_decision_interval operand.
- If the pd_trn_send_decision_intval_sec and the pd_trn_send_decision_interval operands are both specified, the pd_trn_send_decision_intval_sec operand takes precedence.
- If the pd_trn_send_decision_intval_sec and the pd_trn_send_decision_interval operands are both omitted, the following value is assumed depending on the pd_sysdef_default_option operand value:

pd_sysdef_default_option operand value	Default value
recommendable	Default value of the pd_trn_send_decision_intval_sec operand (15 seconds)
v6compatible or v7compatible	Default value of the pd_trn_send_decision_interval operand (5 minutes)

40) pd_trn_send_decision_retry_time = *maximum-wait-time-for-transaction-auto-decision*

~ <unsigned integer>((0-65535)) <<360>> (minutes)

This operand is applicable to a HiRDB/Parallel Server.

Specifies the maximum amount of time to wait for a decision completion notice to be returned after an automatic decision instruction has been sent to a branched transaction. If no decision completion notice is returned within the amount of time specified here, a communication error is considered to have occurred, the decision instruction to the branched transaction is canceled, and the branching-source transaction is decided.

When 0 is specified for this operand, time monitoring is not performed.

Notes

Y must be specified in the pd_trn_rerun_branch_auto_decide operand, or the pd_trn_rerun_branch_auto_decide operand must be omitted.

41) pd_trn_watch_time = *maximum-communication-wait-time-during-transaction-synchronization-point-processing*

~ <unsigned integer>((0, 300-65535)) <<3600>> (seconds)

This operand is specific to a HiRDB/Parallel Server.

Specifies the maximum amount of time to wait for receiving communication (prepare, commit instruction, or response) between transaction branches during transaction synchronization point processing executed in the HiRDB server process. If no instruction or response is received within the specified time, the applicable transaction is rolled back if it has not completed the first phase of a two-phase commit. If the first phase has already been completed, the transaction is completed as specified.

Advantage

Even when a HiRDB client halts the transaction determination instruction (by forcibly terminating a client process, for example), the transaction execution continues unless it is stopped by the HiRDB server. Consequently, locked resources in a database, for example, may be monopolized for a long time. Specifying this operand can shorten the time during which locked resources are monopolized.

Specification guidelines

Normally, you need not specify this operand. Specify it in the following cases:

- The `KFPA11989-E` or `KFPA11722-E` message is output during commit.
- Processing of a `COMMIT` statement takes a long time, even though the number of database updates in the transaction is small.

Operand rules

- If `0` is specified, the receiving wait time is not monitored.
- If a value between `1` and `299` is specified, it is rounded up to `300`.

Note

For the commit instruction in the second phase of a two-phase commit, or for a response, the specification value of this operand is valid only when `pd_dbsync_point=commit` is specified.

42) pd_trn_commit_optimize = ONEPHASE | NOUSE

This operand applies only to a HiRDB/Parallel Server.

Specifies whether to use one-phase commit in a HiRDB/Parallel Server's commitment control. For details about one-phase commitment, see the manual *HiRDB Version 8 Description*.

`ONEPHASE`:

Uses *one-phase commit* for commitment control when the number of branches to be updated within a transaction is one (the number of servers to be updated by one transaction is one). Note that using one-phase commit in commitment control is called *one-phase optimization*.

`NOUSE`:

Uses *two-phase commit* for commitment control. One-phase commit is not used.

Notes

1. Specification of this operand is invalid if an OLTP system has specified two-phase commit.
2. The default value of this operand is `NOUSE` if `v6compatible` is specified for the `pd_sysdef_default_option` operand.The default value of this operand is `ONEPHASE` if recommendable or `v7compatible` is specified for the `pd_sysdef_default_option` operand.
3. If a recovery-unnecessary front-end server is used, the restrictions on this front-end server takes precedence over the specification for this operand. These restrictions are described below.

   The log to be output to a front-end server is suppressed.

   In a recovery-unnecessary front-end server, a UAP that uses the X/Open XA interface to make connection cannot be executed.

2.3.8 Operands related to SQL optimization

43) pd_optimize_level = *SQL-optimization-option* [,*SQL-optimization-option*]...

～ <identifier or unsigned integer>

Specifies SQL optimization options. For details about the SQL optimization options, see the *HiRDB Version 8 UAP Development Guide.* The SQL optimization option facilities are explained as follows.

SQL optimization option facility	Identifier	Unsigned integer	S	P
Forced nest-loop-join	`"FORCE_NEST_JOIN"`	4	Y	Y
Making multiple SQL objects	`"SELECT_APSL"`	10	Y	Y

SQL optimization option facility	Identifier	Unsigned integer	S	P
Increasing the target floatable servers (back-end servers for fetching data)	"FLTS_INC_DATA_BES"	16	N	Y
Prioritized nest-loop-join	"PRIOR_NEST_JOIN"	32	Y	Y
Increasing the number of floatable server candidates	"FLTS_MAX_NUMBER"	64	N	Y
Priority of OR multiple index use	"PRIOR_OR_INDEXES"	128	Y	Y
Group processing, ORDER BY processing, and DISTINCT set function processing at the local back-end server	"SORT_DATA_BES"	256	N	Y
Suppressing use of AND multiple indexes	"DETER_AND_INDEXES"	512	Y	Y
Rapid grouping facility	"RAPID_GROUPING"	1024	Y	Y
Limiting the target floatable servers (back-end servers for fetching data)	"FLTS_ONLY_DATA_BES"	2048	N	Y
Separating data collecting servers	"FLTS_SEPARATE_COLLECT_SVR"	2064	N	Y
Suppressing index use (forced table scan)	"FORCE_TABLE_SCAN"	4096	Y	Y
Forcing use of multiple indexes	"FORCE_PLURAL_INDEXES"	32768	Y	Y
Suppressing creation of update-SQL work tables	"DETER_WORK_TABLE_FOR_UPDATE"	131072	Y	Y
Deriving search acceleration condition	"DERIVATIVE_COND"	262144	Y	Y
Applying key condition that includes scalar operation	"APPLY_ENHANCED_KEY_COND"	524288	Y	Y
Facility for batch acquisition from functions provided by plug-ins	"PICKUP_MULTIPLE_ROWS_PLUGIN"	1048576	Y	Y

S: HiRDB/Single Server

P: HiRDB/Parallel Server

Y: Specification is valid.

N: Specification is not applicable.

Operand specification methods

Select the SQL optimization options to be applied and specify their identifiers or unsigned integers. Although either identifiers or unsigned integers (computed values) can be used to specify options, the use of identifiers is usually recommended.

- Using identifiers

  To apply "forced nest-loop-join" and "making multiple SQL objects," specify the following:

  ```
  pd_optimize_level="FORCE_NEST_JOIN","SELECT_APSL"
  ```

- Using unsigned integers

  To apply "forced nest-loop-join" and "making multiple SQL objects," specify the following:

  ```
  pd_optimize_level=4,10
  ```

- When you have upgraded from HiRDB Version 5.0 or an older version

  The total value specified in HiRDB Version 5.0 or an older version is also valid. If there is no need to change the optimization option, you need not change the specification of this operand after upgrading to HiRDB Version 6 or a newer version. To add the optimization option, use the following example.

  Example:

  Forced nest-loop-join and *making multiple SQL objects* have been applied to HiRDB Version 5.0, and *priority of OR multiple index use* is to be newly added.

  ```
  pd_optimize_level = 14,128
  ```

  However, because this specification makes it difficult to identify the facilities being applied, changing to specification of identifiers is recommended.

Operand rules

- Identifiers and unsigned integers cannot be specified together.
- Identifier specification
  - Enclose each SQL optimization option in quotation marks (").
  - If no SQL optimization option is used, specify NONE. When NONE and an identifier are both specified, the specification of NONE is invalid.
  - Identifiers can be specified in uppercase or lowercase letters.
  - Specifying the same identifier more than once is the same as specifying it once.
- Unsigned integer specification
  - If you do not use the SQL optimization option explained here, specify

`0`. However, if both `0` and a non-zero unsigned integer are specified, the specification of `0` is ignored.

- Specifying the same unsigned integer more than once is the same as specifying it once.

Specification guidelines

For specification guidelines, see *SQL optimization option* in the *HiRDB Version 8 UAP Development Guide*.

Operand default value

If this operand is omitted, the following values are assumed:

- HiRDB/Single Server

  `"PRIOR_NEST_JOIN","PRIOR_OR_INDEXES","DETER_AND_INDEXES","RAPID_GROUPING","DETER_WORK_TABLE_FOR_UPDATE","APPLY_ENHANCED_KEY_COND"`

- HiRDB/Parallel Server

  `"PRIOR_NEST_JOIN","PRIOR_OR_INDEXES","SORT_DATA_BES",`

  `"DETER_AND_INDEXES","RAPID_GROUPING","DETER_WORK_TABLE_FOR_UPDATE","APPLY_ENHANCED_KEY_COND"`

Notes

- If the SQL optimization option is specified in an SQL statement, the SQL optimization specification takes precedence over the specification in this operand. For details on SQL optimization specification, see the manual *HiRDB Version 8 SQL Reference*.
- If the SQL optimization option is specified in an SQL statement (`CREATE PROCEDURE`, `CREATE TYPE`, `ALTER PROCEDURE`, `CREATE TRIGGER`, `ALTER ROUTINE`, or `ALTER TRIGGER`) inside a stored routine or trigger, the SQL optimization option inside the SQL statement takes precedence over the specification in this operand.

Relationship to other operands

- When the `pd_floatable_bes` or `pd_non_floatable_bes` operand is specified in the front-end server definition, specification of "increasing the target floatable servers (back-end servers for fetching data)" and "limiting the target floatable servers (back-end servers for fetching data)" is invalid.
- When `KEY` is specified for the `pd_indexlock_mode` operand, specification of "suppressing creation of update-SQL work tables" is

invalid.

Relationship to client environment definition

The value of this operand can be different for each client. To change the value for a client, specify the `PDSQLOPTLVL` operand in the client environment definition. For details about the `PDSQLOPTLVL` operand, see the *HiRDB Version 8 UAP Development Guide.*

44) pd_additional_optimize_level = *SQL-extension-optimizing-option* [*,SQL-extension-optimizing-option*]...

~ <identifier or unsigned integer> <<'COST_BASE_2'>>

Specifies SQL extension optimizing options. For details about the SQL extension optimizing options, see the *HiRDB Version 8 UAP Development Guide.*

The SQL extension optimizing option facilities are described below.

SQL extension optimizing option facility	Identifier	Unsigned integer
Application of optimizing mode 2 based on cost	`"COST_BASE_2"`	1
Hash execution of a hash join or a subquery[1]	`"APPLY_HASH_JOIN"`	2
Deterring of foreign server execution in SQL statement containing join[1, 2]	`"DETER_JOIN_SQL"`	67108864
Forcing of foreign server execution in SQL statement containing direct product[1, 2]	`"FORCE_CROSS_JOIN_SQL"`	134217728
Deterring of derivation of search acceleration condition that is generated unconditionally and that can be executed in foreign server[1, 2]	`"DETER_FSVR_DERIVATIVE_COND"`	1073741824

[1] These items are valid when *application of optimizing mode 2 based on cost* is specified.

[2] These items are valid for searching a foreign table. Otherwise, they are invalid.

Operand specification methods

Select the SQL extension optimization options to be applied and specify their identifiers or unsigned integers. Although either identifiers or unsigned integers (computed values) can be used to specify options, the use of identifiers is usually recommended.

- Using identifiers

  To apply "application of optimizing mode 2 based on cost" and "hash execution of a hash join or a subquery," specify the following:

```
pd_additional_optimize_level= "COST_BASE_2",
"APPLY_HASH_JOIN"
```

- Using unsigned integers

  To apply "application of optimizing mode 2 based on cost" and "hash-execution of a hash join or a subquery," specify the following:

  ```
  pd_additional_optimize_level=1,2
  ```

Operand rules

- Identifiers and unsigned integers cannot be specified together.
- Identifier specification

  - Enclose each SQL extension optimizing option in quotation marks (").
  - Specify `NONE` if the SQL extension optimizing option explained here is not used. However, if both `NONE` and an identifier other than `NONE` are specified, the specification of `NONE` is invalid.
  - Identifiers can be specified in uppercase or lowercase letters.
  - Specifying the same identifier more than once is the same as specifying it once.

- Unsigned integer specification

  - Specify `0` if the SQL extension optimizing option explained here is not used. However, if both `0` and an unsigned non-zero integer are specified, the specification of `0` is invalid.
  - Specifying the same unsigned integer more than once is the same as specifying it once.

Specification guidelines

For specification guidelines, see *SQL extension optimization option* in the *HiRDB Version 8 UAP Development Guide*.

Notes

- If SQL optimization is specified in an SQL statement, the SQL optimization specification takes precedence over the specification in this operand. For details on SQL optimization specification, see the manual *HiRDB Version 8 SQL Reference*.
- If the SQL extension optimizing option is specified in an SQL statement (`CREATE PROCEDURE`, `CREATE TYPE`, `CREATE TRIGGER`, `ALTER PROCEDURE`, `ALTER ROUTINE`, or `ALTER TRIGGER`) inside a stored

routine or trigger, the SQL extension optimizing option inside the SQL statement takes precedence over the specification in this operand.

Relationship to client environment definition

The value of this operand can be different for each client. To change the value for a client, specify the `PDADDITIONALOPTLVL` operand in the client environment definition. For details about the `PDADDITIONALOPTLVL` operand, see the *HiRDB Version 8 UAP Development Guide.*

45) pd_hashjoin_hashing_mode = <u>TYPE1</u> | TYPE2

Specifies the hashing method to be used when `hash join, subquery hash execution` is specified for the SQL extension optimizing option.

`TYPE1`:

This hashing method is older than Version 07-02.

`TYPE2`:

Provides more uniform hashing than `TYPE1`.

Specification guidelines

- Normally, specify `TYPE2`. However, uniform hashing may not occur depending on the data in the column specified for the join condition. In this case, specify `TYPE1`.
- If specifying `TYPE1` does not produce the expected performance in a user system that has been upgraded to 07-02 or a later version, specify `TYPE2`.

Relationship to client environment definition

The value of this operand can be changed for each client. To change the operand for a client, specify the `PDHJHASHINGMODE` operand in the client environment definition. For details about the `PDHJHASHINGMODE` operand, see the *HiRDB Version 8 UAP Development Guide*.

46) pd_hash_table_size = *hash-table-size*

～ <unsigned integer><<256>> (KB)

- 32-bit mode: **((128-524288))**
- 64-bit mode: **((128-2097152))**

Specifies the size of the hash table to be used when "application of hash-execution of a hash join or a subquery" is specified as an SQL optimization option.

Specification guidelines

For details about the hash table size to be specified in this operand, see the

HiRDB Version 8 UAP Development Guide.

Operand rules

Specify this value as a multiple of 128. If a value that is not a multiple of 128 is specified, the specified value is rounded up automatically to the next multiple of 128.

Relationship to client environment definition

The value of this operand can be different for each client. To change the value for a client, specify the PDHASHTBLSIZE operand in the client environment definition. For details about the PDHASHTBLSIZE operand, see the *HiRDB Version 8 UAP Development Guide.*

47) pd_work_table_option = *work-table-processing-option*

~ <unsigned integer> <<1>>

Specifies the HiRDB processing method to be used for executing an SQL statement that uses a work table. Use the following formula to determine the value to be specified for this operand:

Work table processing option = $a + b$

Work table processing classification	Variable name	Value	Processing method of HiRDB
Lock acquisition method when AND multiple indexes are used[1]	*a*	0	When the use of AND multiple indexes is selected as the access path, the following occurs: If any of the predicates (data) containing the columns for which the indexes to be used for search are defined satisfies the search condition, that data is locked in the shared mode (PR). Consequently, even if WITH EXCLUSIVE LOCK is specified as the lock option[2], a lock is applied in the shared mode (PR), and not in the exclusive mode (EX).
		1	When the use of AND multiple indexes is selected as the access path, the following occurs: If any of the predicates (data) containing the columns for which the indexes to be used for search are defined satisfies the search condition, that data is locked in the specified mode. However, until the search for the first piece of data is completed, a predicate is logically computed for the column for which the index used for the search is defined, and the lock is released from any data that does not satisfy the search condition.

Work table processing classification	Variable name	Value	Processing method of HiRDB
Suppression of message output during automatic extension of the work table buffer[3]	*b*	0	During automatic extension of the work table buffer, the KFPH29008-I message is output. This message is output during the first automatic extension of the work table buffer of each server process.
		8	During automatic extension of the work table buffer, the KFPH29008-I message is not output.

[1] Specifies the HiRDB processing method to be used when using AND multiple indexes.

Use of AND multiple indexes means the following: When a search condition contains multiple conditions connected using AND, and different indexes are defined for each column (for example, `SELECT ROW FROM T1 WHERE C1 = 100 AND C2 = 200`), the multiple indexes are used to create work tables of rows that satisfy the conditions and obtain the product set of these tables.

[2] If no lock option is specified for an SQL statement, `WITH SHARE LOCK` is normally assumed. However, note that the lock option to be assumed differs in the following cases:

- The `FOR UPDATE` clause is specified for the cursor.
- The `PDISLLVL` operand is specified in the client environment definition.
- A data guarantee level is specified for a procedure, facility, or `CREATE PROCEDURE`, `CREATE TRIGGER`, `CREATE TYPE`, `ALTER PROCEDURE`, `ALTER ROUTINE`, or `ALTER TRIGGER` trigger.

[3] As a precondition, the `pd_work_buff_expand_limit` operand must be specified. For automatic extension of the work table buffer, see the description of this operand.

Specification guidelines

Work table processing classification	Value	Specification guidelines
Lock acquisition method when AND multiple indexes are used	0	When 0 is specified, no lock release is performed, and thus the SQL processing time decreases accordingly. However, if multiple users simultaneously try to update the same table using AND multiple indexes, deadlock may occur. To avoid deadlock, specify 1.
	1	When 1 is specified, lock release is performed, and thus the SQL processing time increases accordingly. However, because the ultimate lock range becomes narrower, the concurrent executability of update-UAPs improves.

Work table processing classification	Value	Specification guidelines
Suppression of extension allocation message for work table buffer	`0`	When `0` is specified, you can monitor whether extension has been allocated for the work table buffer.
	`8`	By specifying `8`, you can reduce the messages output volume.

Notes

Note the following when you specify a lock acquisition mode using AND multiple indexes for work table processing classification:

1. The lock acquisition mode using AND multiple indexes that is applied during stored routine and trigger generation (execution of `CREATE PROCEDURE`, `CREATE TRIGGER`, `CREATE TYPE`, `ALTER PROCEDURE`, `ALTER ROUTINE`, or `ALTER TRIGGER`) is applied to procedures, facilities, and triggers. The lock acquisition mode using AND multiple indexes that is applied during the execution of a stored routine (execution of a `CALL` statement) or trigger is not applied.
2. If AND multiple indexes are suppressed by the following operands or options, the specification of the `pd_work_table_option` operand is invalid.

- `pd_optimize_level` operand
- `PDSQLOPTLVL` operand in the client environment definition
- SQL optimization option of `CREATE PROCEDURE`, `CREATE TRIGGER`, `CREATE TYPE`, `ALTER PROCEDURE`, `ALTER ROUTINE`, or `ALTER TRIGGER`

2.3.9 Operands related to narrowed retrieval

48) pd_max_list_users = ***number-of-users-who-can-own-lists-concurrently***

～ <unsigned integer>((0-32767)) <<0>>

Specifies the maximum number of users who can own lists concurrently.

Relationship to other operands

When this operand is specified, the `pd_max_list_count` operand must also be specified.

Notes

- Do not specify an unnecessarily large value for this operand. Increasing the specification value of this operand increases the size of the shared memory used by HiRDB. For details on the shared memory used by servers, see the *HiRDB Version 8 Installation and Design Guide*.

- If you specify the maximum value (`32767`) for both this operand and the `pd_max_list_count` operand, the size of the shared memory may exceed the upper limit of the size that can be allocated by the OS.

49) pd_max_list_count = *number-of-lists-created-per-user*

~ <unsigned integer>((0-32767)) <<0>>

Specifies the maximum number of lists that one user can create.

Relationship to other operands

When this operand is specified, the `pd_max_list_users` operand must also be specified.

Notes

- Do not specify an unnecessarily large value for this operand. Increasing the specification value of this operand increases the size of the shared memory used by HiRDB. For details on the shared memory used by servers, see the *HiRDB Version 8 Installation and Design Guide*.
- If you specify the maximum value (`32767`) for both this operand and the `pd_max_list_users` operand, the size of the shared memory may exceed the upper limit of the size that can be allocated by the OS.

50) pd_list_initialize_timing = <u>INITIAL</u> | DEFER | STANDBY

Specifies the list initialization (deletion) timing. Normally, lists are initialized when HiRDB is started (including restart). You can use this operand to change the initialization timing. If you create a large number of lists, you may want to consider changing the specification value of this operand.

For details on changing the list initialization timing, see the *HiRDB Version 8 System Operation Guide*.

`INITIAL`:

Lists are initialized during HiRDB startup. HiRDB is started only after all lists have been initialized.

`DEFER`:

Lists are not initialized during HiRDB startup. They are initialized when the `ASSIGN LIST` statement is executed. Consequently, there is some overhead associated with the execution of the `ASSIGN LIST` statement. Consider specifying `DEFER` in the following cases:

- You want to shorten the HiRDB startup time.

- You want to shorten the system switchover time (when user server hot standby or the rapid system switchover facility is being used).

`STANDBY`:

This specification becomes valid when the rapid system switchover facility is used. Even if `STANDBY` is specified when the rapid system switchover facility is not being used, `INITIAL` is used as the default.

Lists are initialized when starting the standby HiRDB. They are not initialized during a system switchover. Lists are not initialized when the `ASSIGN LIST` statement is executed following a system switchover, either. Note that you must create the same lists for both the running system and the standby system.

2.3.10 Operands related to system monitoring

51) pd_utl_exec_time = *utility-execution-monitoring-time*

～ <unsigned integer>((0-35791394)) <<0>> (minutes)

Specifies the monitoring time (in minutes) for monitoring the execution time of the following utilities:

- Database load utility (`pdload` command)
- Database reorganization utility (`pdrorg` command)
- Free page release utility (`pdreclaim` command)
- Global buffer residence utility (`pdpgbfon` command)

If the execution of a utility is not terminated within the monitoring time specified by this operand, the executing utility is forcibly terminated, and the error information for identifying the cause of no response is output as shown in the following table.

Error information obtained	Output destination
• Core files • `.deb` files	These files are output in `$PDDIR/spool/save` of the server machine on which the utility was running.

Error information obtained	Output destination
• `pdls -d rpc -a` command execution result • `pdls -d lck` command execution result	The results are output in `/tmp` of the server machine into which the utility command was input. However, if HiRDB executes the command, these results are also output in `$PDDIR/spool/save`. The file names are as follows: *command-nameYYYYMMDDHHMMSSpid*`.txt` • *pid* is a process ID. • If the `pdreclaim` or `pdpgbfon` command is executed, HiRDB internally executes the `pdrorg` command, and therefore `pdrorg` is used as the command name to be assigned to the file name.

Advantage

Even if a utility ceases to respond because of an error (communication or disk error, for example) that occurs during utility execution in a nighttime batch job, the succeeding jobs can continue running.

Specification guidelines

- This operand is designed to deal with no-response errors, and not to monitor for large transactions. Therefore, specify a value for this operand that is somewhat greater than the actual maximum execution time of utilities. For example, if the maximum execution time of the database load utility is approximately 60 minutes, and the maximum execution time of the database reorganization utility is approximately 90 minutes, specify `pd_utl_exec_time=120` by leaving some room. In this case, when a process that is normally terminated in 90 minutes does not return a response even after 30 more minutes, it is determined that a no-response error has occurred.
- If this operand is omitted or `0` is specified for it, a utility may go into a no-response state when the following errors occur. Therefore, specify a value other than `0`.
  - Communication error (including temporary error) between servers
  - Process not responding due to a disk error, for example

Operand rules

- If this operand is omitted or `0` is specified for it, utility execution time is not monitored.
- If monitoring time is specified in the `exectime` operand of the `option` control statement of each utility, the specification in the `exectime` operand takes precedence.

52) pd_watch_time = *SQL-maximum-execution-time*

~ <unsigned integer>((0-65535)) <<0>> (seconds)

This operand is applicable to a HiRDB/Parallel Server.

Specifies a maximum execution time for SQL statements that are executed in a HiRDB server process.

If execution of an SQL is not completed within the specified amount of time, execution of that SQL is terminated.

Advantage

If the HiRDB server does not halt execution of an SQL statement, even though the HiRDB client has canceled SQL execution (by forcibly terminating a client process, for example), the HiRDB server may continue to execute the SQL statement and may lock resources for a long time. Specifying this operand places a limit on such a lock time.

Specification guidelines

Specify the largest value among the following time values:

Time specified by the `PDCWAITTIME` operand of the client environment definition

Time specified by the `pd_lck_wait_timeout` operand

Processing time of the SQL statement with the longest execution time

Notes

- When `0` is specified for this operand, the SQL execution time is not monitored.
- When `0` (no time monitoring) is specified for the `PDCWAITTIME` operand of the client environment definition for an SQL statement, the execution time of that SQL statement is not monitored.
- When a value shorter than the SQL time is specified for this operand, processing may terminate during SQL execution, and an SQL error may be reported to the HiRDB client or abnormal termination may be reported to the HiRDB server.
- For a HiRDB/Single Server, SQL maximum execution time is not monitored, even if this operand is specified. If this operand is specified, the specified value is used as the default value of the `pd_lck_wait_timeout` operand. Therefore, Hitachi recommends that you omit this operand for a HiRDB/Single Server.

53) pd_queue_watch_time = *message-queue-monitoring-time*

~ <unsigned integer>((0-3600)) <<600>> (seconds)

This operand is designed to prevent a HiRDB process from not responding. For details about a server process that has stopped responding, see the *HiRDB Version 8 System Operation Guide*.

Reference note:

The number of HiRDB server processes is restricted by the following operands:

- `pd_max_server_process`

  If a large number of servers are running within a unit, carefully estimate the value to be specified for this operand. Additionally, if the standby-less system switchover facility is used, the estimated value must also take into account system switchovers.

- `pd_max_bes_process`

  If a multiple front-end server or the standby-less system switchover (1:1) facility is used, carefully estimate the value to be specified for this operand.

- `pd_max_dic_process`

  If a multiple front-end server is used, carefully estimate the value to be specified for this operand.

- `pd_ha_max_server_process`

  If the standby-less system switchover (effects distributed) facility is used, carefully estimate the value to be specified for this operand.

- `pd_max_users`

  If the number of concurrent connections is large, an appropriate value must be specified for this operand.

HiRDB uses a message queue for allocating server processes. When a server process stops responding, messages cannot be extracted from the message queue. If messages cannot be extracted from the message queue within the time specified by this operand (message queue monitoring time), a warning message or error message (`KFPS00888-W` or `KFPS00889-E`) is output. This facility is called the message queue monitoring facility. If one of these messages is output, the server process may have stopped responding.

If `0` is specified in this operand, the message queue is not monitored.

For details on the message queue monitoring facility, see the *HiRDB Version 8 System Operation Guide*.

Operand rule

Because message queue monitoring is carried out in 10-second increments, specify for this operand a multiple of 10 seconds (100 or 110 seconds, for example). If the specified value is not a multiple of 10 seconds, the last digit is rounded up. For example, if you specify 105 seconds, 110 seconds is used.

Relationship to other operands

This operand is related to the `pd_queue_watch_timeover_action` operand.

54) pd_queue_watch_timeover_action = continue | stop

Specifies the processing to be performed by HiRDB when messages cannot be extracted from the message queue within the message queue monitoring time.

`continue`:

HiRDB outputs the warning message (`KFPS00888-W`).

`stop`:

HiRDB outputs the warning and error messages (`KFPS00888-W` and `KFPS00889-E`) and abnormally terminates the unit that has the message queue.

Specification guidelines

- Normally, specify `stop` (or omit this operand). If messages cannot be extracted from the message queue within the message queue monitoring time, the HiRDB server process may have stopped responding. Restarting the unit containing the server process that has stopped responding can sometimes correct the non-responding server process.
- If you do not wish to stop HiRDB, specify `continue`. In this case, transactions cannot be executed on the server that has stopped responding. Other servers can still execute transactions. To correct the non-responding server process, use the `pdcancel` command, for example, to terminate the transactions that were being executed in the server process that stopped responding. Afterwards, identify the cause of the server process no-response and take the necessary action. If there were no transactions, use the OS's `kill` command to terminate the server that stopped responding. Afterwards, identify the cause of the server process no-response and take the necessary action. For details on server process no-response and corrective measures, see the *HiRDB Version 8 System Operation Guide*.

55) pd_down_watch_proc = *upper-limit-for-server-process-abnormal-terminations*[,*monitoring-interval*]

This operand is used for monitoring the number of abnormal terminations of a HiRDB server process. Processes to be monitored are those that are abnormally terminated by PDCWAITTIME over or aborting.

If abnormal terminations of server processes occur frequently, new services may not be accepted. However, because server process abnormal termination does not cause HiRDB abnormal termination, HiRDB is in an online stopped state, in effect. When this operand is specified, you can pull HiRDB out of this state by restarting it.

***upper-limit-for-server-process-abnormal-termination*: ~ <unsigned integer>((0-65535))<<0>>**

If abnormal terminations of server processes exceed the value specified in this operand, HiRDB (applicable unit for a HiRDB/Parallel Server) is abnormally terminated. This is called the facility for monitoring abnormal process terminations. For details on this facility, see the *HiRDB Version 8 System Operation Guide*.

For a HiRDB/Single Server, abnormal terminations of single server processes are counted. In the case of a HiRDB/Parallel Server, the total of the abnormal terminations in the front-end servers, back-end servers, and dictionary servers inside the unit is counted.

If 0 is specified, abnormal terminations of server processes are not monitored.

***monitoring-interval*: ~ <unsigned integer>((10-3600))<<600>> (units: seconds)**

Specifies the interval (in seconds) for monitoring abnormal terminations of server processes.

For example, if 100 is specified, abnormal terminations of server processes are monitored every 100 seconds.

Advantages

- Restart of HiRDB refreshes memory and resource statuses, improving the processing efficiency.
- If abnormal termination of server processes occurs frequently, HiRDB is abnormally terminated, and thus the system can be switched over immediately.

Notes

- When a server process is abnormally terminated, the KFPS01820-E message is output. Although this message is also output when the server process is abnormally terminated by the pdcancel command, this is not counted as an abnormal termination.
- For a mutual system switchover configuration, multiple HiRDBs are

activated on the same server machine when system switchover occurs. As a result, the system traffic may increase, causing an adverse effect instead. Therefore, if you specify this operand, Hitachi recommends that you restart HiRDB in the system that was abnormally terminated.

Operand rule

A monitoring interval cannot be specified alone. It must be specified with an upper limit for server process abnormal terminations.

Remarks

- If HiRDB is abnormally terminated by the facility for monitoring abnormal process terminations, the KFPS01821-E and KFPS00729-E messages are output.
- The following table shows the causes of server process abnormal termination and the server processes that are included in the abnormal termination count.

Cause of server process abnormal termination	Inclusion in abnormal termination count			
	Single server process	Front-end server process	Dictionary server process	Back-end server process
PDCWAITTIME operand value of the client environment definition has been exceeded	Y	Y	N[1]	N[1]
pdcancel command	N	N[2]	N	N
Internal forced termination (HiRDB internally issues SIGKILL and terminates a process)	Y[3]	Y[3]	N[1]	N[1]
Abort	Y	Y	Y	Y
Rollback has occurred in a UAP with XA connection	Y	Y	N	N
Abnormal termination of process other than those described here	Y	Y	Y	Y

Legend:

Y: Included in abnormal termination count

N: Not included in abnormal termination count

[1] If an error is detected in a transaction branch, the abnormal terminations of the front-end server process that has occurred in the same transaction branch are counted.

[2] If the pdcancel command is used to forcibly terminate a back-end server process or dictionary server process, the front-end server process is internally and forcibly

terminated. In this case, the abnormal termination of the front-end server process may be counted in some cases.

[3] If an error is detected in a global transaction by an OLTP system, the abnormal terminations of the single server process or front-end server process that have occurred in the same global transaction are counted.

56) pd_host_watch_interval = *host-to-host-monitoring-interval*

~ <unsigned integer>((1-180))<<10>>(Seconds)

Specifies in seconds the amount of time between monitoring of the operating status of other hosts (server machines).

Notes

- This operand is not applicable to a HiRDB/Single Server or to a HiRDB/Parallel Server that has only one unit.
- If this value is too large, it will be difficult to detect errors at other hosts.

Relationship to other operands

This operand is related to the `pd_ipc_conn_nblock_time` operand.

57) pd_watch_resource = MANUAL | AUTO

Specifies whether to output a warning message when the resource usage reaches or exceeds 80%.

`MANUAL`: Do not issue the warning message.

`AUTO`: Issue the warning message.

The following table lists the resources that are monitored.

Monitored resource	Message that is output
Maximum number of concurrent connections specified by the `pd_max_users` operand	`KFPS05123-W`
Concurrently accessible base tables count specified by the `pd_max_access_tables` operand	
Maximum number of RDAREAs specified by the `pd_max_rdarea_no` operand	
Maximum number of HiRDB files comprising an RDAREA specified by the `pd_max_file_no` operand	
Usage of HiRDB file system area for work table files, as specified by the `pdwork` operand	
Maximum number of users who can own lists concurrently, as specified by `pd_max_list_users`	
Maximum number of lists that a user can create, as specified by `pd_max_list_count`	
Number of audit trail files that cannot be specified as swappable targets	
Number of lists created within a server	`KFPH22023-W`

Relationship to other operands

When `AUTO` is specified, 80 is set at the default value for the operands listed below, which eliminates the need to specify these operands (however, if it is necessary to specify a value other than 80 for any of the listed operands, the appropriate value can be specified):

- `pd_max_users_wrn_pnt`*
- `pd_max_access_tables_wrn_pnt`
- `pd_max_rdarea_no_wrn_pnt`
- `pd_max_file_no_wrn_pnt`
- `pdwork_wrn_pnt`
- `pd_max_list_users_wrn_pnt`
- `pd_max_list_count_wrn_pnt`
- `pd_aud_file_wrn_pnt`*
- `pd_rdarea_list_no_wrn_pnt`*

* 50 is set as the trigger for resetting the warning message output status.

Note

After a version upgrade, the number of resources to be monitored may increase (i.e., other operands in addition to those listed above may be included).Therefore, to keep the monitoring statuses up to date, the operands listed above should be specified in order to monitor the individual resources, rather than specifying this operand.

58) pd_max_users_wrn_pnt = ***trigger-for-outputting-warning-message-related-to-number-of-connections-to-HiRDB-server* [*,trigger-for-resetting-warning-message-output-status*]**

trigger-for-outputting-warning-message-related-to-number-of-connections-to-HiRDB-server ~ **<unsigned integer>((0-100))<<0 or 80>>(%)**

Specifies when to output the warning message when the number of connections to the HiRDB server reaches or exceeds a predetermined percentage (percentage of the maximum number of concurrent connections specified by the `pd_max_users` operand). For example, if `200` and `90` are specified for the `pd_max_users` and `pd_max_user_wrn_pnt` operands, respectively, the `KFPS05123-W` warning message is output when the number of connections to the HiRDB server reaches or exceeds 180.

Notes

- When `0` is specified, no warning message is issued.
- Specification of this operand is invalid when `9` or less is specified for the `pd_max_users` operand.

Relationship to other operands

- When this operand is omitted and `MANUAL` is specified for the `pd_watch_resource` operand (or the `pd_watch_resource` operand is omitted), 0 is assumed for the `pd_max_users_wrn_pnt` operand (i.e., no warning message is issued).
- When this operand is omitted and `AUTO` is specified for the `pd_watch_resource` operand, `80` is assumed for the `pd_max_users_wrn_pnt` operand (i.e., the warning message is issued when the number of concurrently executing users reaches 80% of the specified maximum).

trigger-for-resetting-warning-message-output-status ~ **<unsigned integer>((0-99))(%)**

Specifies the trigger for resetting the warning message output status. When the warning message (`KFPS05123-W`) is output, HiRDB goes into the warning message output status. Once HiRDB goes into this status, the warning message is not output again even if the number of connections to the HiRDB server exceeds the warning message output trigger level again. However, when the number of connections to the HiRDB server falls below the trigger for resetting the warning message output status specified here, the warning message output status is reset.

For example, if `pd_max_users_wrn_pnt=90,70` is specified, the warning message is output when the number of connections to the HiRDB server reaches or exceeds 90% of the maximum number of concurrent connections. Afterwards, no warning message is output until the number of connections to the HiRDB server falls below 70% of the maximum number of concurrent connections. After the percentage falls below 70%, and when it subsequently reaches or exceeds 90% again, the warning message is output.

Notes

- When this specification is omitted, warning-message-output-trigger - `30` is assumed as the default (if the result is a negative number, 0 is used).
- If a value greater than the warning message output trigger is specified, the warning message output trigger value is used.

59) pd_max_access_tables_wrn_pnt = ***trigger-for-issuing-concurrently-accessible-base-tables-count-warning***

~ <unsigned integer>((0-100))<<0 or 80>>(%)

Specifies as a percentage of the maximum number of concurrently accessible base tables (as specified by the `pd_max_access_tables` operand) the number of base tables actually being accessed concurrently at which a warning message is to be issued. For example, if 90 is specified for this operand and 200 has been specified for the `pd_max_access_tables` operand, a warning message will be issued when the number of concurrently accessed base tables reaches 180 (90% of 200).`KFPS05123-W` is issued as the warning message.

Note

When `0` is specified, no warning message is issued.

Relationship to other operands

- When this operand is omitted and `MANUAL` is specified for the `pd_watch_resource` operand (or the `pd_watch_resource` operand is omitted), `0` is assumed for the `pd_max_access_tables_wrn_pnt` operand (i.e., no warning message is issued).
- When this operand is omitted and `AUTO` is specified for the `pd_watch_resource` operand, 80 is assumed for the `pd_max_access_tables_wrn_pnt` operand (i.e., the warning message is issued when the number of concurrently accessed base tables reaches 80% of the specified maximum).

60) pd_max_rdarea_no_wrn_pnt = *trigger-for-issuing-RDAREAs-count-warning*

~ <unsigned integer>((0-100))<<0 or 80>>(%)

Specifies as a percentage of the maximum number of RDAREAs (as specified by the `pd_max_rdarea_no` operand) the actual number of RDAREAs at which a warning message is to be issued. For example, if `90` is specified for this operand and `200` has been specified for the `pd_max_rdarea_no` operand, a warning message will be issued when the number of RDAREAs reaches 180 (90% of 200). `KFPS05123-W` is issued as the warning message.

Note

When `0` is specified, no warning message is issued.

Relationship to other operands

- When this operand is omitted and `MANUAL` is specified for the `pd_watch_resource` operand (or the pd_watch_resource operand is omitted), `0` is assumed for the `pd_max_rdarea_no_wrn_pnt` operand (i.e., no warning message is issued).
- When this operand is omitted and `AUTO` is specified for the `pd_watch_resource` operand, `80` is assumed for the `pd_max_rdarea_no_wrn_pnt` operand (i.e., the warning message is issued when the number of RDAREAs reaches 80% of the specified maximum).

61) pd_max_file_no_wrn_pnt = *trigger-for-issuing-HiRDB-files-count-warning*

~ <unsigned integer>((0-100))<<0 or 80>>(%)

Specifies as a percentage of the maximum number of files comprising an RDAREA (as specified by the `pd_max_file_no` operand) the number of files actually comprising an RDAREA at which a warning message is to be issued. For example, if 90 is specified for this operand and 200 has been specified for the `pd_max_file_no` operand, a warning message will be issued when the number of files comprising an RDAREA reaches 180 (90% of 200). `KFPS05123-W` is issued as the warning message.

Note

When `0` is specified, no warning message is issued.

Relationship to other operands

- When this operand is omitted and `MANUAL` is specified for the `pd_watch_resource` operand (or the `pd_watch_resource` operand is omitted), `0` is assumed for the `pd_max_file_no_wrn_pnt` operand (i.e., no warning message is issued).
- When this operand is omitted and `AUTO` is specified for the

pd_watch_resource operand, 80 is assumed for the pd_max_file_no_wrn_pnt operand (i.e., the warning message is issued when the number of files comprising an RDAREA reaches 80% of the specified maximum).

62) pdwork_wrn_pnt = *trigger-for-issuing-work-table-files-warning*

~ <unsigned integer>((0-100))<<0 or 80>>(%)

Specifies as a percentage of the applicable usage* of a HiRDB file system area for work table files (as specified by the pdwork operand) the point at which a warning message is to be issued. For example, if 90 is specified for this operand, a warning message will be issued when the applicable usage rate of a HiRDB file system area for work table files reaches 90%. KFPS05123-W is issued as the warning message.

* This operand specifies a percentage of usage in terms of the items listed below, which are specified with the pdfmkfs command when the HiRDB file system areas are created; the warning message can be issued for each of these items:

- Usage relative to capacity
- Usage relative to the maximum number of files
- Usage relative to the maximum number of increments

Note

When 0 is specified, no warning message is issued.

Relationship to other operands

- When this operand is omitted and MANUAL is specified for the pd_watch_resource operand (or the pd_watch_resource operand is omitted), 0 is assumed for the pdwork_wrn_pnt operand (i.e., no warning message is issued).
- When this operand is omitted and AUTO is specified for the pd_watch_resource operand, 80 is assumed for the pdwork_wrn_pnt operand (i.e., the warning message is issued when the usage reaches 80%).

63) pd_max_list_users_wrn_pnt = *trigger-for-issuing-warning-about-number-of-users-who-have-created-lists*

~ <*unsigned* integer>((0-100)) <<0 or 80>> (%)

Specifies as a percentage of the number of users who can create lists (as specified by the pd_max_list_users operand) the point at which the number of users actually using lists is to cause a warning message to be issued. For example, if 90 is specified for this operand and 200 was specified for the pd_max_list_users operand, a warning message will be issued whenever the number of users using lists reaches 180. KFPS05123-W is issued as the warning message.

Notes

- When 0 is specified, no warning message is issued.
- The warning message is output only once. However, if the applicable server is restarted after the warning message has been output, a check of list usage is executed again, and the warning message will be output again.

Relationship to other operands

- When this operand is omitted and MANUAL is specified for the pd_watch_resource operand (or the pd_watch_resource operand is omitted), 0 is assumed for the pd_max_list_users_wrn_pnt operand (i.e., no warning message is issued).
- When this operand is omitted and AUTO is specified for the pd_watch_resource operand, 80 is assumed for the pd_max_list_users_wrn_pnt operand (i.e., the warning message is issued when the usage reaches 80%).

64) pd_max_list_count_wrn_pnt = *trigger-for-issuing-warning-about-number-of-lists-created-by-a-user*

~ <unsigned integer>((0-100)) <<0 or 80>> (%)

Specifies as a percentage of the number of lists that can be created per user (as specified by the pd_max_list_count operand) the point at which the number of lists created by a user is to cause a warning message to be issued. For example, if 90 is specified for this operand and 200 was specified for the pd_max_list_count operand, a warning message will be issued when a user creates 180 lists. KFPS05123-W is issued as the warning message.

Notes

- When 0 is specified, no warning message is issued.
- The warning message is output only once. However, if the applicable server is restarted after the warning message has been output, a check of list usage is executed again, and the warning message will be output again.
- If multiple users are creating lists simultaneously, the warning message may be output at the server multiple times, depending on the timing.

Relationship to other operands

- When this operand is omitted and MANUAL is specified for the pd_watch_resource operand (or the pd_watch_resource operand is omitted), 0 is assumed for the pd_max_list_count_wrn_pnt operand (i.e., no warning message is issued).

- When this operand is omitted and AUTO is specified for the pd_watch_resource operand, 80 is assumed for the pd_max_list_count_wrn_pnt operand (i.e., the warning message is issued when the usage reaches 80%).

65) pd_rdarea_list_no_wrn_pnt = *trigger-for-issuing-warning-about-number-of-lists-created-at-server* [,*trigger-for-resetting-warning-output-status*]

trigger-for-issuing-warning-about-number-of-lists-created-at-server **~ <unsigned integer>((0-100)) <<0 or 80>> (%)**

Specifies as a percentage of the number of lists that can be created at a server the point at which a warning message is to be issued. For example, if 90 is specified for this operand and the maximum number of lists that can be created at the server is 1000, a warning message will be issued whenever 900 lists have been created. KFPH22023-W is issued as the warning message.

Note

When 0 is specified, no warning message is issued.

Relationship to other operands

- When this operand is omitted and MANUAL is specified for the pd_watch_resource operand (or the pd_watch_resource operand is omitted), 0 is assumed for the pd_rdarea_list_no_wrn_pnt operand (i.e., no warning message is issued).
- When this operand is omitted and AUTO is specified for the pd_watch_resource operand, 80 is assumed for the pd_rdarea_list_no_wrn_pnt operand (i.e., the warning message is issued when the usage reaches 80%).

trigger-for-resetting-warning-output-status **~ <unsigned integer>((0-99)) (%)**

Specifies the trigger for resetting the warning message output status.

Whenever the KFPH22023-W warning message is issued, HiRDB goes into warning message output status. Once HiRDB is in this status, the warning message will not be issued again even when the number of created lists again reaches the trigger value. However, when the number of created lists falls below the trigger for resetting the warning message output status specified here, the warning message output status is reset.

For example, if pd_rdarea_list_no_wrn_pnt=90,70 is specified, the warning message will be issued when the number of created lists reaches 90% of the maximum permissible number of lists. Once this occurs, the warning message will not be eligible to be issued again until the number of created lists falls below 70% of the maximum number of permissible lists allowed. Thereafter, the warning message will be issued if the number of lists again reaches the 90% trigger value.

Notes

- If no value is specified for the trigger for resetting the warning output status, `30` is assumed. If a negative is specified, `0` is assumed.
- If the value specified here is greater than the value specified for the trigger for issuing warning, the value specified here is ignored and the same value as was specified for the trigger for issuing warning is assumed as the trigger for resetting the warning output status.

2.3.11 Operands related to SQL runtime warning output facility

66) pd_cwaittime_wrn_pnt =
output-condition-for-SQL-runtime-warning-information (% specification) |
output-condition-for-SQL-runtime-warning-information (time specification)

Specify this operand when using the SQL runtime warning output facility. You can specify this operand using one of the following two methods:

- Specifying a percentage
- Specifying a time duration

For details about the SQL runtime warning output facility, see the *HiRDB Version 8 System Operation Guide*.

output-condition-for-SQL-runtime-warning-information (% specification): **~ <unsigned integer>((0-99)) or <unsigned decimal number>((0-99.999999))<<0>> (%)**

Specifies the condition for outputting SQL runtime warning information of the SQL runtime warning output facility as a percentage of the maximum client wait time (value of the `PDCWAITTIME` operand in the client environment definition). After SQL execution, HiRDB checks the SQL execution time. If this SQL execution time is determined to exceed the time specified by this operand, the following warning information is output. This is called the *SQL runtime warning output facility*.

- SQL runtime warning information file
- Warning message (`KFPA20009-W`)

Operand specification method

Specify this operand as a percentage (%) of the value of the `PDCWAITTIME` operand. For example, if `100` (seconds) is specified for the `PDCWAITTIME` operand and `90` (%) is specified for the `pd_cwaittime_wrn_pnt` operand, HiRDB checks the SQL execution time after SQL execution. If the determined SQL execution time is 90 seconds or longer, but less than 100 seconds, the warning information is output.

Example

```
PDCWAITTIME = 100
pd_cwaittime_wrn_pnt = 90
```

***output-condition-for-SQL-runtime-warning-information (time specification)*:**
~ <unsigned decimal number>((0-65534.999999))<<0 sec>> (seconds)

Specifies the condition for outputting SQL runtime warning information of the SQL runtime warning output facility as a time duration. After SQL execution, HiRDB checks the SQL execution time. If this SQL execution time is determined to exceed the time specified by this operand, the following warning information is output. This is called the *SQL runtime warning output facility*.

- SQL runtime warning information file
- Warning message (`KFPA20009-W`)

Operand specification method

Specify the time duration (in seconds) to be used as the output trigger. (You can specify up to the sixth decimal.) Add `sec` to the specified value.

Example

```
pd_cwaittime_wrn_pnt = 0.001sec
```

The following explanation is the same for both percentage and time duration specification.

Operand rule

If `0` or `0sec` is specified in this operand, no warning information is output. (The SQL runtime warning output facility is not used.)

Relationship to client environment definition

You can change the value of this operand for each client. To change it for each client, specify the `PDCWAITTIMEWRNPNT` operand of the client environment definition. For details on the `PDCWAITTIMEWRNPNT` operand, see the *HiRDB Version 8 UAP Development Guide*.

Relationship to other operands

This operand is related to the following operands.

- `pd_cwaittime_report_dir`
- `pd_cwaittime_report_size`

67) pd_cwaittime_report_dir = *SQL-runtime-warning-information-file-output-destination-directory*

~ <pathname>

Specifies an absolute path name as the output destination directory for the SQL runtime warning information file. Two SQL runtime warning information files (`pdcwwrn1` and `pdcwwrn2`) are created under the directory specified here.

If this operand is omitted, no SQL runtime warning information file is output. However, the warning message (`KFPA20009-W`) is still output.

For a HiRDB/Parallel Server, the SQL runtime warning information files are output to the server machine containing the front-end server to which the UAP that issued the warning target SQL is connected.

Operand rules

- Specify no more than 255 characters for the path name.

68) pd_cwaittime_report_size = *SQL-runtime-warning-information-file-maximum-size*

~ <unsigned integer>((2048-2147473627))<<100000>>(Bytes)

Specifies the maximum size for the SQL runtime warning information file. The value specified in this operand indicates the size of a single SQL runtime warning information file. Therefore, be careful about the value you specify for this operand because two SQL runtime warning information files are created. For example, if you specify `10,000`, two files, each with a maximum size of 10,000 bytes, are created under the directory.

Specification guidelines

Use the following formula as a guideline when determining the value to be specified for this operand.

{1280 + *SQL-statement-size* (bytes)} × *number-of-pieces-of-warning-information-to-be-stored-in-file*

If a comment or SQL optimization is specified for the SQL statement, also include the size of the comment and the specified SQL optimization size (bytes) in the SQL statement size.

Remarks

- If the volume of data that is output to an SQL runtime warning information file exceeds the value specified by this operand, the output destination is switched to the other file. The two files are used alternately as this process is repeated. During this process, the oldest information is deleted from the switching destination file.

- If the volume of the SQL runtime warning information that is output at one time exceeds the file size, not all of the SQL runtime warning information is output. Only the information that fits in the file size is output. In this case, the number symbol [#] is added to the end of the SQL runtime warning information.

2.3.12 Operands related to the facility for output of extended SQL error information

69) pd_uap_exerror_log_use = YES | NO

Specifies whether to use the facility for output of extended SQL error information. For details about this facility, see the *HiRDB Version 8 UAP Development Guide*.

`YES`:

Uses the facility for output of extended SQL error information. SQL error information is output in an error log file and an SQL error report file.

`NO`:

The facility for output of extended SQL error information is not used.

Relationship to client environment definition

The value of this operand can be changed for each client. To change the operand for a client, specify the `PDUAPEXERLOGUSE` operand in the client environment definition. If both this operand and the `PDUAPEXERLOGUSE` operand in the client environment definition are specified, the `PDUAPEXERLOGUSE` operand takes precedence.

For details about the `PDUAPEXERLOGUSE` operand, see the *HiRDB Version 8 UAP Development Guide*.

70) pd_uap_exerror_log_dir = *SQL-error-report-file-storage-directory*

~ <path name of up to 255 characters>

Specifies an absolute path name for the directory in which to store SQL error report files.

Two SQL error report files are created in the specified directory. Their file names are `pduaperrlog1` and `pduaperrlog2`.

If this operand is omitted, no SQL error information is output in an SQL error report file.

71) pd_uap_exerror_log_size = *SQL-error-report-file-maximum-size*

~ **<unsigned integer>((2048-2147483647))<<1000000>>(bytes)**

Specifies the maximum size of an SQL error report file. The value specified by this operand applies to each of the two SQL error report files that are to be created. When the volume of data that is output to an SQL error report file exceeds the value specified by this operand, the output destination is switched to the other file. The two files are used alternately as this process is repeated. If the volume of the SQL error information that is output at one time exceeds the value specified by this operand, the first through the [*specified value* -1]-th bytes (up to the 999999th byte if 1000000 is specified for this operand) of SQL error information is output. In this case, the hash mark [`#`] is added to the end of the SQL error information.

Specification guidelines

Determine the value to be specified for this operand by taking into consideration the volume of SQL error information that should be retained. You can use the following computation formula:

(*A* + *B*) × *volume-to-be-retained*

- *A* = `1100` + *SQL-statement-size* (bytes)

  This is the size of each piece of SQL error information, excluding the parameter information output size. If an SQL statement contains a comment or the description of SQL optimization specification, the size of the comment or SQL optimization specification also must be included in the SQL statement size. For details about comments and SQL optimization specification, see the manual *HiRDB Version 8 SQL Reference*.

- B = (↑ `pd_uap_exerror_log_param_size`-*operand-value* ÷ 16 ↑ + 1) × 89 × *parameter-count*

  This is the parameter information output size.

72) pd_uap_exerror_log_param_size = ***maximum-data-size-of-parameter-information-to-be-output-to-error-log-file-and-SQL-error-report-file***

~ **<unsigned integer>((0-32008))<<0>>(bytes)**

Specifies the maximum data size for the parameter information to be output to an error log file and an SQL error report file.

- When `1` or a value greater than 1 is specified

  Parameter information is output to an error log file and an SQL error report file.

- When `0` is specified

Parameter information is not output to an error log file or an SQL error report file.

If the parameter information is in the variable-length character string type, BLOB type, or BINARY type, the data size area also is included in the specified value.

If the size of the parameter information to be output to an error log file and an SQL error report file exceeds the value specified for this operand, only the parameter information that fits in the file size is output, and the remainder is discarded.

Relationship to client environment definition

The value of this operand can be changed for each client. To change the operand for a client, specify the `PDUAPEXERLOGPRMSZ` operand in the client environment definition. If both this operand and the `PDUAPEXERLOGPRMSZ` operand are specified, the `PDUAPEXERLOGPRMSZ` operand in the client environment definition takes precedence.

For details about the `PDUAPEXERLOGPRMSZ` operand, see the *HiRDB Version 8 UAP Development Guide*.

2.3.13 Operands related to the SQL reserved word deletion facility

73) pd_delete_reserved_word_file = ***SQL-reserved-word-deletion-file-name-1[,SQL-reserved-word-deletion-file-name-2]...***

~ <identifier of up to 8 characters>

To use the SQL reserved word deletion facility, specify this operand. For this operand, specify the name of an SQL reserved word deletion file. Create the SQL reserved word deletion file under `$PDDIR/conf/pdrsvwd`. For details about the SQL reserved word deletion facility, see the *HiRDB Version 8 UAP Development Guide*.

Operand rules

- For the name of the SQL reserved word deletion file, specify any alphanumeric character string (of up to 8 characters) that begins with a letter.
- You can specify a maximum of 16 SQL reserved word deletion file names.

Format of the SQL reserved word deletion file

***reserved-word*[*,reserved-word*...]**

- When specifying multiple reserved words, use a comma (`,`) as the delimiter. Do not insert a space or tab before or after the comma.
- The maximum length of each definition line is 80 characters. If the

definition exceeds 80 characters, use a line feed. However, do not use a line feed in the middle of a reserved word (keyword).

- You can use both lower- and upper-case letters to specify reserved words.
- For details about SQL reserved words, see the manual *HiRDB Version 8 SQL Reference*.

Note

If you delete a reserved word, you can no longer use an SQL function that uses that reserved word. For details about the reserved words that can be deleted using the reserved word deletion facility, see the manual *HiRDB Version 8 SQL Reference*.

Relationship to client environment definition

Specify any SQL reserved word deletion file that is used during UAP execution in the `PDDELRSVWDFILE` operand in the client environment definition. For details about the `PDDELRSVWDFILE` operand, see the *HiRDB Version 8 UAP Development Guide*.

2.3.14 Operands related to lock

74) pd_lck_deadlock_info = <u>Y</u> | N

Specifies whether or not deadlock information and timeout information are to be output. These types of information are output to a directory named `$PDDIR/spool/pdlckinf`. For details about the deadlock and timeout information, see the *HiRDB Version 8 System Operation Guide*.

`Y`: Output deadlock information and timeout information.

`N`: Do not output deadlock information or timeout information.

75) pd_lck_wait_timeout = *lock-release-wait-time*

~ <unsigned integer>((0-65535))<<Max(180, pd_watch_time value)>>(seconds)

Specifies in seconds the maximum amount of time to wait for lock release (the maximum amount of time in lock release wait status).This is the elapsed time from when a lock release request is placed in wait status until it is released.

If the wait status is not released within the specified amount of time, the SQL will return an error. When `0` is specified, lock wait time will not be monitored and waiting will continue until the wait status is released.

Note

If this operand is omitted, the value of the `pd_watch_time` operand may be used as the default value in some cases. Although the pd_watch_time operand is invalid, even if it is specified for a HiRDB/Single Server, the specified value may be used as the default value for the pd_lck_wait_timeout operand in some cases. Therefore, Hitachi recommends that you omit the pd_watch_time operand for a HiRDB/ Single Server.

76) pd_lck_release_detect = interval | pipe

Specifies the method to be used by HiRDB to detect lock release (method of detecting whether or not the process that locked a resource has released that lock).

interval:

Determine the lock release status by checking the lock management area at a regular interval.

pipe:

Use a pipe file (process-to-process communication pipe) to receive the lock release notice.

Specification guidelines

Following are guidelines for specifying this operand:

Specified value	HiRDB processing	Advantages and application criteria
interval	Determines whether the lock has been released by checking the lock management area in the shared memory. HiRDB checks the lock management area at a regular interval; this interval is specified by the pd_lck_release_detect_interval operand.	Even if the process that has locked a resource releases the lock, the release will not be detected until the next time the lock management area is checked. Consequently, a UAP that has a fast processing time per transaction may end up waiting for a long time. However, this wait does not place any load on the CPU or open any file. If a small value is specified for the pd_lck_release_detect_interval operand, the CPU usage may increase too much, adversely affecting the throughput. Specifying interval has the effect of reducing the CPU load when a slow CPU is being used.

Specified value	HiRDB processing	Advantages and application criteria
pipe	Uses a pipe file to determine whether or not the lock has been released. The process that has locked a resource sends a lock release notice to the process that is waiting for release. When the process that has locked the resource releases it, the process that is waiting for the release can detect the release. Because a maximum of one file descriptor is used per process, it is necessary to re-estimate the number of file descriptors.	• Throughput improves if the processing time per transaction does not exceed the value specified by the pd_lck_release_detect_interval operand. • CPU workload increases if lock release wait occurs frequently. Additionally, the number of open files is increased because pipe files are opened. • Specify pipe when there are system resources (CPU and file descriptors) that can be spared.

Notes

- When pipe is specified for the first time after HiRDB has been newly installed, the pipe file is initialized during activation of the unit. Consequently, unit activation completion may take longer than expected.
- When pipe has been specified, the pipe file is opened when lock-release wait occurs. Therefore, add to the estimate for the file descriptors for the entire system (nfile in the case of HP-UX) the value determined from the following formula (as applicable):

  HiRDB/Single Server:

  pd_max_users operand value + pd_max_reflect_process_count value

  HiRDB/Parallel Server:

  (a + 3 × *server-count-inside-unit*) × b

  a: Total of the values determined for all the servers inside the unit, computed using the following formulas:

  Back-end server: Value of pd_max_bes_process + pd_max_reflect_process_count

  Dictionary server: Value of pd_max_dic_process + pd_max_reflect_process_count

  If the pd_max_bes_process or pd_max_dic_process operand is omitted, use the value of pd_max_users. If a unit contains multiple back-end servers, compute a value for each back-end server.

  b: 16 (number of threads waiting for lock release)

- When pipe is specified in the Linux version, the pipe file is opened when lock-release wait occurs. In this case, it is necessary to specify the maximum number of open files in the system, including open pipe files, using the method described below.

  Specify the following in `/etc/rc.d/rc.local`:

  `/sbin/sysctl -w fs.file-max =` *maximum-number-of-open-files-in-system*

Relationship to other operands

If `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand, the default value for this operand is `interval`.

77) pd_lck_release_detect_interval = *lock-release-detection-interval*

～ <unsigned integer>((1-1000))<<10>>(Milliseconds)

Specifies the interval at which the lock management area is to be checked.

- When 49 or less is specified

  The interval will begin at the value specified by this operand and thereafter will be 50 milliseconds.

- When 50 or more is specified

  The interval will begin at 50 milliseconds and thereafter will be the value specified by this operand.

Condition

The `interval` option must be specified for the `pd_lck_release_detect` operand.

Specification guidelines

- As a rule, this operand can be omitted. This operand should be specified (instead of using the default value) only when there is a performance problem.
- If too small a value is specified, the CPU workload will increase if lock release waits occur frequently.
- If too large a value is specified, the wait time may increase.
- The value to be specified can be determined by referring to `WAIT TIME` in the statistical information related to system operation from the statistics analysis utility. If the wait time that is output in the statistical information is smaller than the value of this operand, the value of this operand should be reduced.

78) pd_nowait_scan_option = LOCK | <u>NOLOCK</u>

Specifies the processing method to be used when `WITHOUT LOCK NOWAIT` search is executed.

`LOCK`:

When a `WITHOUT LOCK NOWAIT` search is executed, locking is applied on a row-by-row basis. Only those rows that are not yet being updated or have already been updated by other transactions can be referenced. A row that is being updated by another transaction cannot be referenced until the updating process is finished.

`NOLOCK`:

When a `WITHOUT LOCK NOWAIT` search is executed, locking is not applied. Even those rows being updated by other transactions can be referenced. Note, however, that when a row being updated is referenced, it may not be possible to receive the search target row in some cases.

Note

When you specify `LOCK` for this operand and update a non-fixed table or execute a `WITHOUT LOCK NOWAIT` search, the number of locked resources increases by 1 per transaction. Furthermore, the base row log volume for each piece of data that is output by an update operation using an `UPDATE` statement for a non-fixed table increases by 628 bytes.

79) pd_lck_queue_limit = *trigger-for-issuing-warning-about-number-of-users-waiting-for-lock-release*

~ <unsigned integer>((0-500))<<10>>

Specifies the number of users waiting for lock release of a resource at which time a warning message is to be issued. When the number of users who are waiting for the same resource to be released reaches the specified value, the `KFPS00446-W` warning message is issued. This message is output to the message log file and the syslogfile.

When `0` is specified, no warning message is issued.

80) pd_deadlock_priority_use = Y | <u>N</u>

Specifies whether or not deadlock priorities are to be used.

`Y`:

Use priority control to determine the program that is to be executed first when deadlock occurs. When deadlock occurs, the SQL of the program with the lower priority is terminated with an error. If both programs have the same priority, the SQL of the program that executed its transaction most recently is terminated with an error. The deadlock priorities are specified with the

`PDDLKPRIO` operand of the client environment definition. For details about the `PDDLKPRIO` operand of the client environment definition, see the *HiRDB Version 8 UAP Development Guide.*

`N`:

Terminate with an error the SQL of the program that executed its transaction most recently.

81) pd_command_deadlock_priority = 32 | 64 | 96 | 120

Specifies the deadlock priority value of a command.

The value specified for this operand is effective for the following operation commands:

- `pdhold -b` (reference-possible backup hold)
- `pdhold -s` (synchronizing hold)
- `pddbchg` (switching of replica status for a replica RDAREA)
- `pdorbegin` (committing a database for online reorganization)
- `pdorend` (reflection processing of online reorganization)

For details about how to change the deadlock priority value of a command, see the *HiRDB Version 8 System Operation Guide*.

Condition

`Y` must be specified for the `pd_deadlock_priority_use` operand.

Specification guideline

If a deadlock occurs in a system that allows a job program to make a retry, you can increase the priority of the operation command to cause an error in the job program. The smaller the specified value, the higher the priority during a lock.

82) pd_key_resource_type = <u>TYPE1</u> | TYPE2

Specifies the method of creating a locked resource for an index key value. For details about how to create a locked resource for an index key value, see the *HiRDB Version 8 UAP Development Guide.*

`TYPE1`:

Create a locked resource for an index key value by using bit shifts and exclusive-OR operations.

`TYPE2`:

Create a locked resource for an index key value by using byte-order exclusive-OR operations.

Specification guidelines

- As a rule, `TYPE1` is specified. If the key length exceeds 10 bytes, exclusive competition may be caused by synonyms in the locked resource for the index key value. Specification of `TYPE1` can prevent this.
- `TYPE2` is used for HiRDB Version 5.0, 05-02 or versions older than 4.0, 04-05. A user with an upgraded system should specify `TYPE2` if the older method does not cause exclusive competition by synonyms in the locked resource for the index key value.
- `TYPE1` should be specified if the indexed column contains character string data.
- `TYPE2` should be specified if the indexed column contains numeric data.

83) pd_indexlock_mode = {KEY | <u>NONE</u>}

Specifies the locking method for a B-tree index.

`KEY`: Lock using index key value.

`NONE`: Do not lock using index key value (executes *index key value no lock*).

For details about index key value no lock, see the *HiRDB Version 8 UAP Development Guide.*

Relationship to other operands

If the `pd_inner_replica_control` operand is specified, `NONE` is assumed as this operand's value.

2.3.15 Operands related to buffers

84) pd_sql_object_cache_size = *SQL-object-buffer-size*

~ <unsigned integer> <<(pd_max_users value + 3) × 22>>(KB)

- 32-bit mode: **((22-256000))**
- 64-bit mode: **((22-2000000))**

Specifies in kilobytes the size of the buffer area (shared memory) in which SQL objects are to be placed.

Specification guidelines

- SQL objects are saved in a buffer until the user's transaction has terminated. The buffer must be large enough to store the SQL objects of all transactions that will be executed concurrently.
- SQL analysis can be reduced by saving the SQL objects of static SQLs in the buffer after transaction termination (until the buffer runs out of

space) and sharing them among multiple users who execute the same UAP. To effectively utilize the buffer, it should be allocated so that the SQL objects of frequently used UAPs are resident in the buffer.

- To estimate the buffer length, first determine the buffer length needed for UAP execution from the length of the SQL objects from the SQL statements to be issued by the UAP. Then, compute the buffer length by considering the number of UAPs that will be executed concurrently and the number of concurrently executing users.
- For details about how to estimate the length of the SQL object from a single SQL statement, see *D.2 Formulas for determining size of SQL object buffer (pd_sql_object_cache_size).*

Tuning the specified value

For details about how to tune the SQL object buffer length, see the *HiRDB Version 8 System Operation Guide.*

85) pd_def_buf_control_area_assign = <u>INITIAL</u> | TRAN

Specifies whether the control areas for the buffers for table definition information, buffers for view analysis information, buffers for user-defined type information, and buffers for routine definition information are to be allocated in a single batch at the time of HiRDB activation or in a single batch at the time of transaction activation.

`INITIAL`:

Allocate the various buffer control areas in a single batch at the time of HiRDB activation. These control areas are not released until HiRDB terminates.

`TRAN`:

Allocate the various buffer control areas in a single batch at the time of transaction activation. These control areas are released when the transaction is completed.

Specification guidelines

Specify `INITIAL` when performance is important; transaction performance will be improved. However, because a larger shared memory will be required than when `TRAN` is specified, re-evaluate the memory requirements. For details about how to estimate the shared memory size, see the *HiRDB Version 8 Installation and Design Guide.*

86) pd_thread_max_stack_size = *maximum-stack-size-for-use-by-one-thread*

～ <unsigned integer>((*0-maximum stack size permitted by the OS* (up to 2047)))<<0>>(MB)

This operand is applicable only to HiRDB/Parallel Server.

Specifies the maximum stack size to be used by one thread. When `0` is specified or this operand is omitted, there is no limit to the stack size to be used by one thread. Because HiRDB controls pseudo-threads, it allocates as stack memory that it secures on its own. Consequently, the value of the OS's `maxssiz` parameter (stacksize for Solaris and Linux) is invalid for pseudo-threads. This parameter is specified instead of the `maxssiz` parameter.

Specify a value that is equal to or smaller than `maxssiz` for this operand.

Advantages

A UAP error or a complex SQL request from a UAP may result in one thread using a large amount of system resources, causing the OS to hang up. When this operand is specified, it limits the stack size to be used by one thread, thus preventing the OS from hanging up.

Notes

The value specified by this operand must be equal to or smaller than the maximum stack size permitted by the OS. The maxssiz parameter can be used to determine the maximum stack size permitted by the OS.

2.3.16 Operands related to shared memory

87) pd_shmpool_attribute = free | fixed

Specifies whether to fix the shared memory used by the HiRDB unit controller in memory. Note that the AIX 5L version of HiRDB(32-bit mode) does not support fixing of the shared memory, and therefore specification of `fixed` for this operand is invalid.

`free`:

Do not fix the shared memory in memory. Depending on the size of the real memory, another page of shared memory may be created, affecting performance adversely.

`fixed`:

Fix the shared memory in memory.

Advantage

Fixing the shared memory to be used by the HiRDB in the memory ("fixed" specified) prevents shared memory paging, thus improving the access performance to the shared memory.

Specification guidelines

Whether or not the shared memory is to be fixed should be based on the computed shared memory size and the real memory size of the server machine. Because there is a limit to the amount of memory that can be fixed for the HiRDB, fixing too much shared memory may cause frequent paging of other types of memory. How much real memory there is and the amount of memory to be occupied by the shared memory pool need to be carefully evaluated.

For the formula for computing the shared memory sizes to be used by the unit controller and individual servers, see the *HiRDB Version 8 Installation and Design Guide*.

For the formula for computing the shared memory sizes to be used by the unit controller and individual servers, see the *HiRDB Version 8 Installation and Design Guide*.

Notes

The following notes apply to the specification of `fixed` in the AIX 5L version:

- Set operating system parameters for the OS. If the applicable operating system parameters are not set, this operand is ignored. For details about the operating system parameters to be set, see *Specifying parameters unique to AIX 5L* in the *HiRDB Version 8 Installation and Design Guide*.
- Set the server machine's memory and the `VMM` parameter's `maxpin` so that the following condition is satisfied:

  (machine's memory size) x (`maxpin` setting)/100 ≥ $a + b^{\#} + c$

  a: Size of shared memory segment for the unit controller

  b: Sum of the sizes of all shared memory segments for global buffers

  c: Sum of the sizes of all shared memory segments used by the OS that fixes shared memory or the additional program products

  #: Add this value when `fixed` is specified in the `pd_dbbuff_attribute` operand.
- You can use an AIX command to determine whether or not shared memory has been fixed in memory; for details, see the AIX 5L manual.

88) pd_dbbuff_attribute = free | fixed

Specifies whether to fix the shared memory used by the global buffer in the real memory. Note that the AIX 5L version of HiRDB(32-bit mode) or the Linux version of HiRDB does not support fixing of the shared memory, and therefore specification of `fixed` for this operand is invalid.

`free` (default value for 64-bit mode):

Do not fix the shared memory to be used by the global buffer in the real memory.

`fixed` (default value for 32-bit mode):

Fix the shared memory to be used by the global buffer in the real memory.

Advantage

Fixing the shared memory to be used by the global buffer in the real memory ("fixed" specified) prevents shared memory paging, thus improving the performance of accesses to the shared memory.

Specification guidelines

- Whether or not the shared memory is to be fixed should be determined based on the computed shared memory size and the real memory size of the server machine. If a memory page that is quite large relative to the real memory is fixed, the result may be frequent paging or a virtual memory shortage. Therefore, how much real memory there is and the amount of memory to be occupied by the shared memory pool need to be evaluated carefully. For the formula for computing the shared memory size to be used by the global buffer, see the *HiRDB Version 8 Installation and Design Guide*.
- When the size of the shared memory to be page-fixed is subtracted from the size of the real memory, the result should not equal or exceed half the size that is obtained by subtracting the size of the shared memory to be page-fixed from the swap area size.

Relationship to other facilities

- If `fixed` is specified, the shared memory used by a dynamically modified global buffer is also fixed in the real memory. Therefore, carefully consider the real memory size before adding or modifying a global buffer.

Notes

If `fixed` is specified in the AIX 5L version, see the notes for the `pd_shmpool_attribute` operand.

Relationship to other operands

If `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand, the default value is as follows:

- HiRDB in 32-bit mode: `fixed`
- HiRDB in 64-bit mode: `free`

2.3.17 Operands related to message log files

89) pd_mlg_msg_log_unit = manager | local

This operand is applicable only to a HiRDB/Parallel Server.

Specifies a message log output destination unit.

manager:

Outputs message logs in the system manager unit's message log file and syslogfile. The message log file is created under $PDDIR/spool of the system manager's server machine.

local:

Outputs message logs in the message log file and syslogfile of the unit that sent the message. The message log file is created under $PDDIR/spool of each server machine.

Specification guidelines

- To use the system manager unit to centrally manage message logs, specify manager for this operand.
- If an error or other factor has caused the system manager to stop, message logs may be output after a time delay in syslogfile of the unit that sent the message, or messages themselves may not be output at all. If you specify local for this operand, messages can be output to the unit that sent the message without any delay. For details about how to distribute message log output, see the *HiRDB Version 8 System Operation Guide*.

The following table shows the relationship between the value of this operand and the message log output destination.

pd_mlg_msg_log_unit value	Condition	Message log output destination
manager	Normal operation	Message logs are output in the message log file and syslogfile of the server machine where the system manager is located.
	System manager unit error or communication error	Message logs are output in each server machine's syslogfile. Some message logs may not be output. Message logs may be output with a delay.
local	Normal operation	Message logs are output in the message log file and syslogfile of each server machine.
	System manager unit error or communication error	

Relationship to other operands

If you specify `local` for this operand, estimate the size of the message log file again and specify the result for the `pd_mlg_file_size` operand.

90) pd_mlg_file_size = *maximum-message-log-file-size*

～ <unsigned integer>((64-1000000))<<1024>>(KB)

Specifies in kilobytes the maximum size of a message log file.

Two message log files are provided. When the amount of stored message information reaches the size specified here, the message log files are swapped.

If you specify `local` for the `pd_mlg_msg_log_unit` operand, a message log file having the size specified here is created in each server machine.

2.3.18 Operands related to statistical information

91) pd_statistics = Y | N

Specifies whether or not collection of the statistics log is to be started at the time of HiRDB startup.

`Y`:

Begin collecting the statistics log at the time of HiRDB startup.

`N`:

Do not begin collecting the statistics log at the time of HiRDB startup.

When `Y` is specified, statistics log information on the system operation of the overall unit only is collected. If information on each server or information other than the statistics information on system operation is needed, the `pdstend` command must be entered to stop statistical information collection after HiRDB startup has been completed, then the `pdstbegin` command can be entered to begin collecting statistical information again.

When `N` is specified, it is still possible to collect a statistics log during HiRDB operation by entering the `pdstbegin` command.

Notes

The statistics log information cannot be acquired in the following cases:

- The `pdstbegin` operand is specified.
- The standby-less system switchover (1:1) facility is used and the system is switched to an alternate BES unit.
- A unit to which the standby-less system switchover (effects distributed) facility is applied

- No server has started in the unit.
- The log application site uses Real Time SAN Replication based on the log-only synchronous method (the `KFPS04689-W` message is displayed).

92) pd_stj_file_size = *maximum-statistics-log-file-size*

~ <unsigned integer>((64-1000000))<<1024>>(KB)

Specifies in kilobytes the maximum size of a statistics log file.

Two statistics log files are provided. When the amount of stored statistics log information reaches the size specified here, the statistics log files are swapped.

Specification guidelines

- For details about how to determine the value to be specified for this operand, see *D.1 Formulas for determining size of statistics log file (pd_stj_file_size)*.
- Specify this operand such that the following relationship with the `pd_stj_buff_size` operand is maintained:

  `pd_stj_file_size ≥ pd_stj_buff_size × 2`

93) pd_stj_buff_size = *statistics-log-buffer-size*

~ <unsigned integer>((32-512)) <<32>> (KB)

Specifies the statistics log buffer size.

Specification guidelines

The default value of 32 is appropriate when the following types of statistical information are not to be output:

- SQL object execution information
- Statistics about SQL object transmission

If these types of statistical information are to be output, specify the value obtained by adding 32 to the result obtained from the following formula (however, if the result exceeds 512, specify 512):

$(a \div 1024) \times (0.03 \div b)$

a: Statistics log output volume (bytes)

For details, see *D.1 Formulas for determining size of statistics log file (pd_stj_file_size).*

b: Statistical information output time (seconds)

2.3.19 Operands related to RPC trace information

94) pd_rpc_trace = Y | N

Specifies whether or not RPC trace information is to be collected. HiRDB maintenance information is output in the RPC trace information.

Normally, this operand should be omitted.

`Y`: Collect RPC trace information.

`N`: Do not collect RPC trace information.

Note

Specifying `Y` for this operand degrades communication performance.

95) pd_rpc_trace_name = *"name-for-RPC-trace-collection-files"*

~ <pathname of up to 254 characters> <<$PDDIR/spool/rpctr>>

Specifies an `absolute pathname` for the filename for the RPC trace files. Three RPC trace files are created, with 1, 2, and *l* suffixed to the specified file name.

Notes

- Files with a maximum size of `pd_rpc_trace_size value × 2` are created under the directory specified by this operand. Attention should be paid to the file capacity.
- To output RPC trace information to a different path for each server machine when the system common definition file is shared, this operand must be specified in each unit control information definition.

96) pd_rpc_trace_size = *RPC-trace-collection-file-size*

~ <unsigned integer>((1024-2147483648))<<4096>>(Bytes)

Specifies the size of the RPC trace files in bytes.

The value specified by this operand determines the size of both RPC trace file 1 and RPC trace file 2.

Specification guidelines

An RPC trace collects in a single file in time order the information on the communications among all processes. If the volume of this information exceeds the specified value, trace data output is switched to the other file, which means that older trace information will be overwritten. When this happens, the amount of trace information that is available may be inadequate, making troubleshooting difficult. For this reason, at least 1,000,000 should be specified for this operand.

Note

The value specified by this operand does not apply to RPC trace file *l*, because *l*'s size is fixed at 0 bytes.

2.3.20 Operands related to troubleshooting information

97) pd_cancel_dump = put | noput

This operand is designed to reduce the amount of troubleshooting information to be output.

Specifies whether or not troubleshooting information is to be collected in the following cases:

- When a UAP does not terminate within the monitoring time specified by the `PDCWAITTIME` operands in the client environment definition
- When a UAP being executed is canceled by the `pdcancel` command

For details about the troubleshooting information to be collected, see the *HiRDB Version 8 System Operation Guide.*

`put`:

Troubleshooting information is collected. Because the troubleshooting information is output to files under `$PDDIR/spool`, a space shortage may occur in the file system.

Note that the troubleshooting information that has been collected is automatically deleted by HiRDB at the following timings.

- Every 24 hours while HiRDB is running (the deletion interval can be changed using the `pd_spool_cleanup_interval` operand).
- When HiRDB is started (whether to delete the troubleshooting information can be changed using the `pd_spool_cleanup` operand).

If the HiRDB administrator is to delete the troubleshooting information, the `pdcspool` command must be executed.

`noput`:

Do not collect troubleshooting information. Because no troubleshooting information will be collected, the load on the file system is reduced. Specify this option if UAP cancellation occurs frequently during normal operation and there is no need to investigate the causes.

For details about the error information that is displayed in the event of abnormal termination, see Table *2-1 Error information that is displayed in the event of abnormal termination* in the section on the `pd_dump_suppress_watch_time` operand.

Notes

If a UAP is canceled by the `pdcancel` command with the `-d` option specified, troubleshooting information is collected regardless of the option specified for this operand.

98) pd_client_waittime_over_abort = Y | N

This operand is designed to reduce the amount of troubleshooting information to be output.

This operand specifies whether to collect the following types of troubleshooting information when the client maximum wait time (value of the `PDCWAITTIME` operand in the client environment definition) is exceeded during transaction execution.

- Shared memory dump (collected only once)
- Simple dump
- Detailed information inside a unit

Troubleshooting information is collected into `$PDDIR/spool/save`.

`Y`:

Collect troubleshooting information. A shared memory dump is collected only when the maximum wait time of the first client after HiRDB activation is exceeded.

In the case of HiRDB/Parallel Server, troubleshooting information is collected from the back-end server processes and dictionary server processes that are related to the connected front-end server processes. In the case of HiRDB/Single Server, troubleshooting information is collected from the single server process.

`N`:

Do not collect troubleshooting information. In the case of a HiRDB/Parallel Server, troubleshooting information is not collected from the connected front-end server processes. Related back-end server processes and dictionary server processes are rolled back, and no troubleshooting information is collected from them. In the case of a HiRDB/Single Server, no troubleshooting information is collected from the single server process.

Notes

If troubleshooting information is maintained indefinitely in the file, it will use up file space. Note that the troubleshooting information that has been collected is automatically deleted by HiRDB at the following timings.

- Every 24 hours while HiRDB is running (the deletion interval can be changed using the `pd_spool_cleanup_interval` operand).

- When HiRDB is started (whether to delete the troubleshooting information can be changed using the `pd_spool_cleanup` operand).

If the HiRDB administrator is to delete the troubleshooting information, the `pdcspool` command must be executed.

99) pd_dump_suppress_watch_time = *troubleshooting-information-output-suppression-time*

~ <unsigned integer>((0-3600)) <<0>>(Seconds)

This operand is designed to reduce the amount of troubleshooting information to be output.

This operand specifies the amount of time (in seconds) during which to suppress outputting the troubleshooting information again (files under `$PDDIR/spool`) that is output when any of the following situations occurs.

- The time specified in `PDCWAITTIME` is exceeded.
- The UAP being executed is cancelled by the `pdcancel` command (except when the `-d` option is specified).
- A process is abnormally terminated.

Once troubleshooting information is output, no troubleshooting information is output again until the time specified by this operand has elapsed. For example, if `60` is specified for this operand, no troubleshooting information is output again until 60 seconds have passed because troubleshooting information was previously output.

Note that if `0` is specified in this operand, outputting of troubleshooting information is not suppressed.

Advantage

If there are multiple HiRDB server processes, they may be abnormally terminated continuously because of timeout, for example. When abnormal terminations of server processes occur continuously, troubleshooting information, such as core and simple dump, is repeatedly collected, thus causing a space shortage in the disk in which the HiRDB directory is located. If such a shortage occurs, HiRDB may be abnormally terminated. Therefore, specify this operand to make sure that no disk space shortage occurs.

Notes

- If the `-d` option is specified for the `pdcancel` command, if abnormal termination is caused by an internal conflict, or if a signal is received from outside, troubleshooting information is collected regardless of the value specified for this operand.
- Table 2-1 shows the error information that is displayed in the event of

abnormal termination.

Table 2-1: Error information that is displayed in the event of abnormal termination

Cause of abnormal termination	Error information	pd_dump_suppress_watch_time value			
		0		Not 0	
		pd_cancel_dump value			
		put	noput	put	noput
PDSWAITTIME over	Save core file	N	N	N	N
	Debug information file	N	N	N	N
	Simple dump file	N	N	N	N
	KFPA20009-W message	N	N	N	N
	SQL runtime warning information file	N	N	N	N
PDSWATCHTIME over	Save core file	N	N	N	N
	Debug information file	N	N	N	N
	Simple dump file	N	N	N	N
	KFPA20009-W message	N	N	N	N
	SQL runtime warning information file	N	N	N	N

Cause of abnormal termination		Error information	pd_dump_suppress_watch_time value			
			0		Not 0	
			pd_cancel_dump value			
			put	noput	put	noput
PDCWAITTIME over	pd_client_waittime_over_abort=Y	Save core file	Y	Y	Y+	Y+
		Debug information file	Y	Y	Y+	Y+
		Debug information file 2	Y	Y	Y+	Y+
		Simple dump file	Y	Y	Y+	Y+
		KFPA20009-W message	Y	Y	Y+	Y+
		SQL runtime warning information file	Y	Y	Y+	Y+
		Shared memory dump file	F	F	F	F
	pd_client_waittime_over_abort=N	Save core file	N	N	N	N
		Debug information file	Y	N	Y+	N
		Debug information file 2	N	N	N	N
		Simple dump file	Y	N	Y+	N
		KFPA20009-W message	Y	N	Y+	N
		SQL runtime warning information file	Y	N	Y+	N
		Shared memory dump file	N	N	N	N

Cause of abnormal termination		Error information	pd_dump_suppress_watch_time value			
			0		Not 0	
			pd_cancel_dump value			
			put	noput	put	noput
pdcancel command	-d specified	Save core file	Y	Y	Y	Y
		Debug information file	Y	Y	Y	Y
		Simple dump file	Y	Y	Y	Y
		KFPA20009-W message	Y	Y	Y	Y
		SQL runtime warning information file	Y	Y	Y	Y
	-d not specified	Save core file	N	N	N	N
		Debug information file	Y	N	Y+	N
		Simple dump file	Y	N	Y+	N
		KFPA20009-W message	Y	N	Y+	N
		SQL runtime warning information file	Y	N	Y+	N
Internal kill9[1]		Save core file	N	N	N	N
		Debug information file	Y	N	Y+	N
		Simple dump file	Y	N	Y+	N
		KFPA20009-W message	Y	N	Y+	N
		SQL runtime warning information file	Y	N	Y+	N
Internal kill3[2]		Save core file	Y	Y	Y+	Y+
		Debug information file	Y	Y	Y+	Y+
		Simple dump file	Y	Y	Y+	Y+
		KFPA20009-W message	Y	Y	Y+	Y+
		SQL runtime warning information file	Y	Y	Y+	Y+

Cause of abnormal termination	Error information	pd_dump_suppress_watch_time value			
		0		Not 0	
		pd_cancel_dump value			
		put	noput	put	noput
Abort[3]	Save core file	Y	Y	Y+	Y+
	Debug information file	Y	Y	Y+	Y+
	Simple dump file	Y	Y	Y+	Y+
	KFPA20009-W message	Y	Y	Y+	Y+
	SQL runtime warning information file	Y	Y	Y+	Y+
	Abort information file	Y	Y	Y+	Y+
Others[4]	Save core file	D	D	D	D
	Debug information file	Y	Y	Y	Y
	Simple dump file	D	D	D	D
	KFPA20009-W message	D	D	D	D
	SQL runtime warning information file	D	D	D	D

Legend:

Y: Outputs error information. The specification of the pd_dump_suppress_watch_time operand is invalid.

N: Does not output error information.

Y+: Outputs error information. The specification of the pd_dump_suppress_watch_time operand is valid.

D: Error information may not be output depending on how the process is terminated.

F: After the unit is started, error information is output during the first dump.

[1] Issuance of internal SIGKILL, such as abnormal termination of a UAP by OpenTP1. Does not include PDCWAITTIME over or abnormal termination by the pdcancel command.

[2] Issuance of internal SIGQUIT, such as error detection. Does not include

PDCWAITTIME over or abnormal termination by the pdcancel command.

[3] HiRDB detects a conflict and executes abort().

[4] SIGSEGV, SIGBUS, receiving of a signal from outside, exit, and other unexpected errors.

100) pd_debug_info_netstat = Y | N

This operand is designed to reduce the amount of troubleshooting information to be output.

Specifies whether or not troubleshooting information (network information) is to be collected when a HiRDB process or HiRDB (unit) terminates abnormally.

Y: Collect network information.

N: Do not collect network information.

Troubleshooting information related to network information is output to the *server-name-n.deb* file under $PDDIR/spool/save, where *n* is a serial number between 1 and 3. The following are file name examples:

Examples

sdsl.deb

fes2.deb

Specification guidelines

- Normally, this operand need not be specified.
- Specifying N reduces the load on the network associated with collecting network information when a HiRDB process or HiRDB (unit) terminates abnormally. However, it may not be possible to identify the causes of errors.

101) pd_spool_cleanup_interval = *troubleshooting-information-deletion-interval*

~ <unsigned integer>((0-744))<<24>>(Times)

This operand is used for deleting the troubleshooting information and temporary work files that have been output. If these items are not deleted, they may cause a space shortage in the disk in which the HiRDB directory is located. If such a shortage occurs, HiRDB may be abnormally terminated. Therefore, HiRDB regularly deletes the following files:

- Troubleshooting information file (files in $PDDIR/spool)
- Temporary work files (files in $PDDIR/tmp)

This operand specifies the deletion interval (hours). For example, if 48 is specified for this operand, these files are deleted every 48 hours. Normally (if this operand is omitted), files are deleted every 24 hours.

Note that time counting begins when HiRDB is normally started. When HiRDB is normally terminated, time counting also stops. Then, the count returns to 0 during the next normal startup.

Specify the files to be deleted using the `pd_spool_cleanup_interval_level` operand explained as follows.

Operand rule

If `0` is specified, files are not deleted.

Specification guidelines

If `24`, `48`, `72`, and so on are specified for this operand, files are deleted at the predetermined time. Specify the time so that files are deleted during the time period that does not overload the system.

Notes

- Even while HiRDB is stopped because of planned termination, forced termination, or abnormal termination, time counting continues. However, if the deletion time arrives while HiRDB is stopped, files are not deleted. Files are not deleted until the next deletion time. To restart HiRDB after deleting the files, execute the `pdcspool` command.
- If the `TMPDIR` environment variable is specified, the temporary work files used by a command or utility are not output to `$PDDIR/tmp`, but to the directory specified by the `TMPDIR` environment variable. The temporary work files that are output to the directory specified by the `TMPDIR` environment variable are not subject to regular deletion. Therefore, use the `rm` command of the OS, for example, to delete them.

Remarks

The difference between the `pd_spool_cleanup_interval` and `pd_spool_cleanup` operands is as follows:

- The `pd_spool_cleanup_interval` operand is related to regular deletion of troubleshooting information.
- The `pd_spool_cleanup` operand is related to the deletion of troubleshooting information during HiRDB startup.

Therefore, if you plan to run HiRDB continuously for 24 hours, consider specifying the `pd_spool_cleanup_interval` operand. If you plan to terminate HiRDB every day, consider specifying the `pd_spool_cleanup` operand.

102) pd_spool_cleanup_interval_level = *number-of-days* [, *deletion-type*]

This operand is used for deleting the troubleshooting information and temporary work files that have been output, and specifies the condition for regularly deleting the troubleshooting information and temporary work files.

number-of-days: **~ <unsigned integer>((1-24855)) <<7>> (days)**

Troubleshooting information files that are older than the number of days specified here are deleted. For example, if `3` is specified, all troubleshooting information files, except for those created within the last 3 days (or 3 days × 24 hours = 72 hours), are deleted.

deletion-type: **<character string> <<all>>**

Specifies the type of troubleshooting information file to be deleted.

`all`: All files are to be deleted.

`dump`: Only the files internally acquired by HiRDB are to be deleted.

The following are the types of troubleshooting information files that are deleted.

Troubleshooting information file type	Directory name	all	dump	Remarks
Deadlock and timeout information	`pdlckinf`	Y	N	Output when an error occurs during locking.
Access path information	`pdsqldump`	Y	N	Output when the access path display utility is used.
Save core file, etc.	`save`	Y	Y	Output when a process is abnormally terminated.
Shared memory dump file	`pdshmdump`	Y	Y	Output when a process or unit is abnormally terminated.
Simple dump files	`pdsysdump`	Y	Y	None
	`pdsdsdump`	Y	Y	Nonexistent in a HiRDB/Parallel Server

	`pdfesdump`
	`pddicdump`
	`pdbesdump`
		Y	Y	Nonexistent in a HiRDB/Single Server

System log file status information file	`pdjnlinf`	Y	N	Files under `/pdjnlinf/errinf` are not deleted.
Transaction information file	`pdtrninf`	Y	N	Output when Real Time SAN Replication is used.

Y: File is deleted.

N: File is not deleted.

Note

Directory names under `$PDDIR/spool` are shown.

All temporary work files, except for those listed as follows, are deleted regardless of the deletion type specification. Parentheses indicate directory names under `$PDDIR/tmp`.

- Current working directory (`home`) of the process in which HiRDB is to start
- Shared memory information file (`pdommenv`)
- Differential information files of the `pdbufls` command (files with names that begin with `CMb`)

Condition

A value other than `0` must be specified for the `pd_spool_cleanup_interval` operand.

Specification guidelines

Specify a value that is longer than the execution time of commands (including utilities). For example, if the execution of the `pdcopy` command, which collects backup, requires 24 hours (1 day), specify at least 2 for the number of days. If you do not specify a value that is longer than the execution time of the command, the temporary work files being used by the command are deleted, and thus the command may not run correctly.

Operand rule

If you specify a deletion type, you must also specify a number of days.

Remarks

The difference between the `pd_spool_cleanup_interval_level` and `pd_spool_cleanup_level` operands is as follows:

- The `pd_spool_cleanup_interval_level` operand is related to regular deletion of troubleshooting information.
- The `pd_spool_cleanup_level` operand is related to the deletion of troubleshooting information during HiRDB startup.

Therefore, if you plan to run HiRDB continuously for 24 hours, consider specifying the `pd_spool_cleanup_interval_level` operand. If you plan to terminate HiRDB every day, consider specifying the `pd_spool_cleanup_level` operand.

Relationship to other operands

If v6compatible or v7compatible is specified in the pd_sysdef_default_option operand, the default deletion type for this operand is dump.

103) pd_spool_cleanup = normal | force | no

This operand is used for deleting the troubleshooting information that has been output.

Specifies whether or not troubleshooting information files (files under $PDDIR/spool) that were output previously by HiRDB are to be deleted when HiRDB is started. This operand is related to the pd_spool_cleanup_level operand described below.

normal:

Delete the files when HiRDB is started normally or is restarted following a planned termination.

force:

Delete the files whenever HiRDB is started, regardless of the HiRDB activation mode.

no:

Do not delete the files.

Specification guidelines

If troubleshooting information files take up too much disk space, specify normal or force.

Note

Troubleshooting information that is output by a command or utility executed by a user other than the HiRDB administrator may not be deleted. In such a case, a user who is authorized to delete troubleshooting information files will have to use the OS's rm command to delete the files.

Remarks

The difference between the pd_spool_cleanup_interval and pd_spool_cleanup operands is as follows:

- The pd_spool_cleanup_interval operand is related to regular deletion of troubleshooting information.
- The pd_spool_cleanup operand is related to the deletion of troubleshooting information during HiRDB startup.

Therefore, if you plan to run HiRDB continuously for 24 hours, consider specifying the pd_spool_cleanup_interval operand. If you plan to terminate HiRDB every day, consider specifying the pd_spool_cleanup operand.

104) pd_spool_cleanup_level = *number-of-days* [, *deletion-type*]

This operand is used for deleting the troubleshooting information that has been output, and specifies the condition for deleting the troubleshooting information files during HiRDB startup.

number-of-days: **~ <unsigned integer>((0-24855)) <<7>> (days)**

Specifies a number of days when troubleshooting information that is older than the specified number of days is to be deleted. For example, if 3 is specified, all troubleshooting information will be deleted except for the information that is fewer than 3 days old (3 days × 24 hours = 72 hours).

If 0 is specified, all troubleshooting information files are deleted.

deletion-type: **<character string> <<all>>**

Specifies the type of troubleshooting information to be deleted.

all: Delete all file types.

dump: Delete only files collected internally by HiRDB.

The following are the types of troubleshooting information files that are deleted:

Troubleshooting information file type	Directory name	all	dump	Remarks
Deadlock and timeout information	pdlckinf	Y	N	Output when an error occurs during locking.
Access path information	pdsqldump	Y	N	Output when the access path display utility is used.
Save core file, etc.	save	Y	Y	Output when a process is abnormally terminated.
Shared memory dump file	pdshmdump	Y	Y	Output when a process or unit is abnormally terminated.
Simple dump files	pdsysdump	Y	Y	None
	pdsdsdump	Y	Y	Nonexistent in a HiRDB/Parallel Server
	pdfesdump pddicdump pdbesdump	Y	Y	Nonexistent in a HiRDB/Single Server

Troubleshooting information file type	Directory name	all	dump	Remarks
System log file status information file	pdjnlinf	Y	N	Files under /pdjnlinf/errinf are not deleted.
Transaction information file	pdtrninf	Y	N	Output when Real Time SAN Replication is used.

Y: File is deleted.

N: File is not deleted.

Note

Directory names under $PDDIR/spool are shown.

Condition

normal or force (default value) must be specified for the pd_spool_cleanup operand.

Operand rule

A number of days and a deletion type must both be specified.

Remarks

The difference between the pd_spool_cleanup_interval_level and pd_spool_cleanup_level operands is as follows:

- The pd_spool_cleanup_interval_level operand is related to regular deletion of troubleshooting information.
- The pd_spool_cleanup_level operand is related to the deletion of troubleshooting information during HiRDB startup.

Therefore, if you plan to run HiRDB continuously for 24 hours, consider specifying the pd_spool_cleanup_interval_level operand. If you plan to terminate HiRDB every day, consider specifying the pd_spool_cleanup_level operand.

Relationship to other operands

If v6compatible or v7compatible is specified in the pd_sysdef_default_option operand, the default deletion type for this operand is dump.

105) pd_module_trace_max = *maximum-number-of-module-traces-that-can-be-stored*

～ <unsigned integer>((126-16383))<<126>>

A HiRDB process records the history of the executed functions and macros inside the process private memory. This history is called a *module trace*. This operand specifies the number of module trace records. The content of this history is loaded into the `core` file and is output when a process error occurs.

Specification guideline

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

Process private memory of the following size is allocated to each process:

In the 32-bit mode: 64 + 48 × *pd_module_trace_max operand value* (bytes)

In the 64-bit mode: 64 + 64 × *pd_module_trace_max operand value* (bytes)

106) pd_module_trace_timer_level = 0 | 10 | 20

Specifies how to acquire the time to be output in module traces. The following table explains the meaning of the value specified for this operand.

Specified value	Time acquisition method
`0`	Time is output in seconds at every module trace output location.
`10`	Time is output in microseconds only at performance-critical module trace output locations, such as those before and after input/output processing, and time is output in seconds at other locations.
`20`	Time is output in microseconds at every module trace output location.

Specification guideline

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

If you specify a value other than `0` for this operand, a function for acquiring time in microseconds is issued, and as a result, system performance may decline.

2.3.21 Operands related to RDAREAs

107) pd_max_rdarea_no = *maximum-number-of-RDAREAs*

~ <unsigned integer>((5-8388592)) <<200>>

Specifies the maximum number of RDAREAs allowed. If the total number of RDAREAs exceeds the value specified for this operand, HiRDB cannot be started normally. Here, RDAREAs also include master directory RDAREAs, data directory RDAREAs, and data dictionary RDAREAs.

Specification guidelines

- Specify a value that is equal to or greater than the total number of RDAREAs (leave some extra room). You can use the `pddbls` command to check the total number of RDAREAs. If you plan to add RDAREAs, specify a value by taking the additional RDAREAs into account.
- For a HiRDB/Parallel Server, the value of this operand is applied to each back-end server. For example, if `100` is specified for this operand, a maximum of 100 RDAREAs can be created in each back-end server. Therefore, use the largest number of RDAREAs among the back-end servers as the guideline when specifying a value for this operand.
- If you are using the inner replica facility, include the number of replica RDAREAs as well.
- If you are using shared RDAREAs, also add the number of shared RDAREAs used by reference-only back-end servers.

Notes

- Do not specify an unnecessarily large value for this operand. Increasing the specification value of this operand increases the size of the shared memory used by HiRDB. If a shortage occurs in the shared memory, HiRDB may not be able to start.
- Do not specify this operand for a utility special unit.

108) pd_max_file_no =
maximum-number-of-HiRDB-files-comprising-an-RDAREA

~ <unsigned integer>((5-134217728)) <<400>>

Specifies the maximum number of HiRDB files that comprise an RDAREA. If the total number of HiRDB files exceeds the value specified for this operand, HiRDB cannot be started normally. Here, HiRDB files comprising an RDAREA also include the HiRDB files of master directory RDAREAs, data directory RDAREAs, and data dictionary RDAREAs.

Specification guidelines

- Specify a value that is equal to or greater than the total number of HiRDB files comprising an RDAREA (leave some extra room). You can use the `pdfstatfs` command to check the number of HiRDB files inside each HiRDB file system area. If you plan to add HiRDB files, specify a value by taking them into account. HiRDB files are added when RDAREAs are added, reinitialized, or extended.

- For a HiRDB/Parallel Server, the value of this operand is applied to each back-end server. For example, if `100` is specified for this operand, a maximum of 100 HiRDB files can be created in each back-end server. Therefore, use the largest number of HiRDB files among the back-end servers as the guideline when specifying a value for this operand.
- If you are using the inner replica facility, include the number of the HiRDB files comprising replica RDAREAs as well.
- If you are using shared RDAREAs, also add the number of HiRDB files comprising the shared RDAREAs used by reference-only back-end servers.

Notes

- Do not specify an unnecessarily large value for this operand. Increasing the specification value of this operand increases the size of the shared memory used by HiRDB. If a shortage occurs in the shared memory, HiRDB may not be able to start.
- Do not specify this operand for a utility special unit.

109) pd_rdarea_warning_point = *segment-usage-ratio-1*[,*segment-usage-ratio-2*[,*segment-usage-ratio-3*]]

～ <unsigned integer>((0-100))(%)

Specifies as percentages triggers for issuing a warning message (`KFPH00211-I` or `KFPA12300-I`) about RDAREA segment usage.

The following table shows the relationship between whether or not this operand is specified and the segment usage ratio warning message for the different types of RDAREAs.

RDAREA type	pd_rdarea_warning_point not specified	pd_rdarea_warning_point specified
• Master directory RDAREA • Data directory RDAREA	No message is output.	Segment use has started that makes the segment usage ratio equal to the value specified for all segments of the applicable RDAREA.
• Data dictionary RDAREA • User RDAREA	Segment use has started that makes the segment usage ratio 80%, 90%, or 100% in terms of the relative position of the segment in the last file of the applicable RDAREA.	Segment use has started that makes the segment usage ratio equal to the value specified for all segments of the applicable RDAREA.

RDAREA type	pd_rdarea_warning_point not specified	pd_rdarea_warning_point specified
• LOB RDAREA	Segment use has started that makes the segment usage ratio 80%, 90%, or 100% in terms of the relative position of the segment in the last file of the applicable RDAREA.	Segment use has started that makes the segment usage ratio equal to the value specified for all segments of the applicable RDAREA.

The following table shows specification examples of this operand:

Message output condition	Operand specification value
To output a message when segment use has started that makes the segment usage ratio 80%, 90%, or 100% of all the segments in the applicable RDAREA	`pd_rdarea_warning_point = 80,90,100`
To output a message when segment use has started that makes the segment usage ratio 50% or 90% of all the segments in the applicable RDAREA	`pd_rdarea_warning_point = 50,90`
To not output a segment usage warning message	`pd_rdarea_warning_point = 0`

Operand rules

- Up to three values may be specified.
- When the same value is specified more than once, only one message will be output for that value.
- When `0` is specified for all three values, no segment usage warning message is output.
- When `0` and a non-zero numeric value are both specified, the non-zero numeric value is used as the trigger for message output.

Note

Specification of this operand is not applicable to the database initialization utility (`pdinit`) or the database recovery utility (`pdrstr`).The message content is the same regardless of whether or not this operand is specified.

110) pd_rdarea_open_attribute_use = Y | N

Specifies whether to use the `DEFER` or `SCHEDULE` attribute as the RDAREA opening trigger.

`Y`: Uses the `DEFER` or `SCHEDULE` attribute.

`N`: Does not use the `DEFER` or `SCHEDULE` attribute.

When this operand is omitted or when `N` is specified, the RDAREA opening trigger attribute is always INITIAL. Therefore, even if the DEFER or SCHEDULE attribute is specified as the RDAREA opening trigger in the operand or utility described as follows, the specification is invalid.

- pd_rdarea_open_attribute
- Database initialization utility
- Database structure modification utility

Notes

- When Y is specified, HiRDB requires a larger shared memory. Consequently, a shared memory shortage may occur, preventing the HiRDB system from starting.
- If the rapid system switchover facility, standby-less system switchover (1:1) facility or standby-less system switchover (effects distributed) facility is used, Y is assumed for this operand. Because the size of the shared memory used by the server increases as a result, re-estimate the shared memory size. For the formula for estimating the size of the shared memory used by a server, see the *HiRDB Version 8 Installation and Design Guide*.

111) pd_rdarea_open_attribute = INITIAL | DEFER | SCHEDULE

Specifies the standard value for the RDAREA opening trigger attribute.

The attribute specified by this operand is assumed for RDAREAs for which open attribute is not specified by the database initialization utility or the database structure modification utility.

For System RDAREAs, INITIAL is always assumed.

Specification guidelines

- The same attribute should be specified for the RDAREAs in the same HiRDB file system area. If different attributes are specified, the expected result may not be obtained.
- The following table lists the opening/closing triggers for each attribute and their advantages and disadvantages.

Attribute	Initial status	Opening trigger	Closing trigger	Advantage	Disadvantage
INITIAL	Open	• HiRDB startup • pdopen command execution	pdclose command execution	Fast execution from the first SQL.	System startup takes time.

Attribute	Initial status	Opening trigger	Closing trigger	Advantage	Disadvantage
`DEFER`	Closed	• Initial access to RDAREA • `pdopen` command execution	`pdclose` command execution	• Fast system startup. • Regular SQLs are executed at high-speed, as well as after the initial access.	First access to each RDAREA takes time.
`SCHEDULE`	Closed	• Initial RDAREA access inside transaction • `pdopen` command execution	• Transaction termination • `pdclose` command execution	• Fast system startup. • Avoids concentrated file opening.	• Initial access to RDAREA results in high workload for a transaction. • If a hybrid method is used for processing Real Time SAN Replication, an overhead of at least 2 seconds occurs for each transaction during its termination.

- The following table shows the operation mode appropriate to each attribute:

Attribute	Appropriate operation mode
`INITIAL`	HiRDB file system area is opened at system startup and keeps the RDAREA information resident in the memory. The HiRDB file system area is opened also during the initial RDAREA access. Because the RDAREA information is not re-created in this case, high-speed operation can be performed from the first SQL. The initial status of the RDAREA at system startup is open, and this RDAREA status will not change unless an operation command is entered subsequently, except during shift to an error shutdown. *This attribute is recommended unless an unusual operation mode is used.* When this attribute is used, a closed RDAREA cannot be accessed.

Attribute	Appropriate operation mode
`DEFER`	HiRDB file system area is not opened at system startup. Instead, the HiRDB file system area is opened during the initial RDAREA access and the RDAREA information is kept resident in the memory. In the second and subsequent accesses, the processing beyond opening of the HiRDB file system area is not performed, and thus high-speed operations can be achieved. The initial status of the RDAREA at system startup is closed, and each RDAREA is opened during the initial access to that RDAREA. The RDAREA status will not change unless an operation command is entered subsequently, except during a shift to an error shutdown. Specify this attribute if you want to avoid cases in which a large number of HiRDB file system areas are opened concurrently or if you want to shorten the time required for starting HiRDB. When HiRDB is restarted, the RDAREAs to be recovered are opened during recovery processing. When this attribute is used, a closed RDAREA can also be accessed.
`SCHEDULE`	HiRDB file system area is not opened at system startup. Instead, after HiRDB startup, the HiRDB file system area is opened during the initial RDAREA access in each transaction and the RDAREA information is kept resident in the memory. When a transaction terminates, the HiRDB file system area opened in that transaction is closed. Thereafter, the processing beyond opening will also be performed during the initial access to an RDAREA whenever the transaction is changed, and thus the workload required for transactions will increase. The initial status of the RDAREA at system startup is closed, and the RDAREA is kept open only during the transaction for the accessed RDAREA. When a transaction terminates, all RDAREAs that were opened in the transaction are closed. If the `pdopen` command is entered, these RDAREAs can be kept open until they are closed by the next shutdown. It is also possible to use other operation commands to change the status of the RDAREAs. If an error is detected, an error shutdown occurs. *This attribute should be specified when it is necessary to compensate for many HiRDB file system areas being opened simultaneously or when it is necessary to reduce the HiRDB system startup time.* When HiRDB is restarted, the RDAREAs to be recovered are opened during recovery processing and closed after the completion of the recovery processing. When this attribute is used, a closed RDAREA can also be accessed. If a hybrid method is used for processing Real Time SAN Replication, the system waits for synchronization with a database at the remote site. During transaction termination, an overhead of at least *number-of-RDAREAs-accessed* × 2 seconds may occur.

Note

The following table describes the notes related to the use of the rapid system switchover facility, standby-less system switchover (1:1) facility, and standby-less system switchover (effects distributed) facility.

Facility used	Notes
Rapid system switchover facility	A standby unit that is targeted by the rapid system switchover facility has not opened RDAREAs while it is in a standby state. Furthermore, to minimize the time required for system switchover, the standby unit opens only those RDAREAs that are necessary for full recovery when system switchover occurs, and does not open other RDAREAs. Therefore, the RDAREA opening trigger for the standby system cannot be `INITIAL`. The `INITIAL` attribute of RDAREAs is changed to `DEFER`.

Facility used	Notes
Standby-less system switchover (1:1) facility	To minimize the time required for system switchover, the standby-less system switchover (1:1) facility opens only those RDAREAs that are necessary for full recovery when system switchover occurs, and does not open other RDAREAs. Therefore, the opening trigger for the RDAREAs in the normal BES or alternate portion is as follows: • When system switchover occurs, the opening trigger for the RDAREAs in the alternate portion is `SCHEDULE`. • When the error is corrected and the system switches back to the normal BES, the opening trigger for RDAREAs with the `INITIAL` or `DEFER` attribute under the normal BES is changed to `DEFER`. RDAREAs with the `SCHEDULE` attribute retain this attribute.
Standby-less system switchover (effects distributed) facility	To minimize the time required for system switchover, the standby-less system switchover (effects distributed) facility opens only those RDAREAs that are necessary for full recovery when system switchover occurs, and does not open other RDAREAs. Therefore, when system switchover occurs, the opening trigger for RDAREAs with the `INITIAL` attribute under the guest BES is changed to `DEFER`.

The following table shows the relationships between the opening trigger for RDAREAs and the rapid system switchover facility, standby-less system switchover (1:1) facility, and standby-less system switchover (effects distributed) facility.

Condition			pd_rdarea_open_attribute_use specification value			
			N	Y		
				RDAREA opening trigger		
				INITIAL	DEFER	SCHEDULE
System switchover facility not used[1]			`INITIAL`	`INITIAL`	`DEFER`	`SCHEDULE`
Standby system switchover facility[1]	Rapid system switchover facility not used					
	Rapid system switchover facility	Running system				
		Standby system	`DEFER`[2]	`DEFER`	`DEFER`	`SCHEDULE`

Condition			pd_rdarea_open_attribute_use specification value			
			N	Y		
				RDAREA opening trigger		
				INITIAL	DEFER	SCHEDULE
Standby-less system switchover (1:1) facility[1]	Accepting portion (other than the alternate portion of the alternate BES unit)	Running system	INITIAL[2, 3]	INITIAL[3]	DEFER	SCHEDULE
		Standby system	DEFER[2]	DEFER	DEFER	SCHEDULE
	Alternate portion	Running system	SCHEDULE[2]	SCHEDULE	SCHEDULE	SCHEDULE
		Standby system				
Standby-less system switchover (effects distributed) facility[1]	Normal start or restart		INITIAL	INITIAL	DEFER	SCHEDULE
	Restart due to system switchover		DEFER[2]	DEFER	DEFER	SCHEDULE

[1] The opening trigger for the System RDAREA is INITIAL.

[2] It is assumed that pd_rdarea_open_attribute_use = Y is specified.

[3] Changed to DEFER after a restart.

112) pd_shared_rdarea_use = Y | N

This operand applies only to a HiRDB/Parallel Server.

Specifies whether to use a shared RDAREA.

Y:

A shared RDAREA is used.

N:

A shared RDAREA is not used.

Notes

- If you omit this operand (or specify N for it) even though a shared RDAREA has been defined, HiRDB cannot be normally started.
- Specifying Y for this operand allocates the same shared memory block

as that allocated when `commit` is specified for the `pd_dbsync_point` operand.

2.3.22 Operands related to global buffers

113) pd_dbbuff_lru_option = SEPARATE | MIX

Specifies the LRU management method for the global buffer. However, if `commit` is specified for the `pd_dbsync_point` operand, `MIX` is assumed unconditionally.

`SEPARATE`:

> Manage the reference and update buffers with separate LRUs. If a global buffer shortage occurs, the reference buffer in the global buffer that was accessed first is purged from the memory. This option should be specified when the number of references and updates per transaction is relatively small, as in an online job.

`MIX`:

> Manage all global buffers with a single LRU. If a global buffer shortage occurs, the buffer in the global buffer that was accessed first is purged from the memory.
>
> This option should be specified when both a large number of retrievals and a large number of updates occur, as when an online job and a batch job coexist.

For details about the LRU management methods for the global buffer, see the *HiRDB Version 8 Description.*

Relationship to other operands

> If `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand, the default value for this operand is `SEPARATE`.

114) pd_dbbuff_modify = Y | N

Specifies whether the global buffer is dynamically modified by the `pdbufmod` command while HiRDB is running. For details on dynamic modification of the global buffer, see the *HiRDB Version 8 System Operation Guide*.

`Y`: Dynamically modifies the global buffer.

`N`: Does not dynamically modify the global buffer.

Condition

> If `Y` is specified for this operand (to dynamically modify the global buffer), HiRDB Advanced High Availability is required.

Relationship to other operands

This operand is related to the following operands.

- `SHMMAX`
- `pdbuffer`
- `pd_max_add_dbbuff_no`
- `pd_max_add_dbbuff_shm_no`

115) pd_dbbuff_lock_release_detect = pipe | interval

When the global buffer is accessed, a global buffer lock acquisition process occurs. This operand specifies the method that the process that is waiting for lock acquisition uses for detecting the lock release of the global buffer. This operand affects the processing performance when the access frequency to the same global buffer is high. The following table shows the processing mode for each operand specification and its characteristics.

Specification	Processing by HiRDB	Characteristics	
		Throughput	CPU usage
`pipe`	The lock-releasing process uses a pipe file to inform the process waiting for lock acquisition of lock release	Low	Low
`interval`	The process waiting for lock acquisition checks whether lock is still on at regular intervals.	High	High

Note

Throughput and CPU usage indicate general trends only. They may vary depending on the execution environment and the value specified for the `pd_dbbuff_lock_interval` or `pd_dbbuff_lock_spn_count` operand.

Specification guidelines

Determine the value for this operand by referring to *Lock release contention wait generation rate in global buffer lock release processing* in the *HiRDB Version 8 System Operation Guide*.

Relationship to other operands

If `interval` is specified for this operand, check the values specified for the following operands.

- `pd_dbbuff_lock_spn_count`
- `pd_dbbuff_lock_interval`

116) pd_dbbuff_lock_spn_count =

number-of-spins-during-lock-acquisition-wait-processing

~ <unsigned integer>((0-2147483646))<<100>>

Specifies the number of spins during lock acquisition wait processing when `interval` is specified for the `pd_dbbuff_lock_release_detect` operand.

An overview of the lock acquisition processing for the global buffer is provided as follows. This operand specifies how many times step 1 is repeated.

1. If the lock for the global buffer is released, the lock is acquired. If the lock is successfully acquired, the process is terminated. However, if the lock cannot be acquired, the process is repeated up to the number of times specified for this operand.
2. If acquisition fails in step 1, the system sleeps for the amount of time specified in the `pd_dbbuff_lock_interval` operand (waits for the specified amount of time to elapse).
3. Returns to step 1.

Condition

`interval` must be specified for the `pd_dbbuff_lock_release_detect` operand.

Specification guidelines

Determine the value for this operand by referring to *Lock release contention wait generation rate in global buffer lock release processing* in the *HiRDB Version 8 System Operation Guide*.

Relationship to other operands

If this operand is specified, check the specification of the `pd_dbbuff_lock_interval` operand.

117) pd_dbbuff_lock_interval = *interval-during-lock-acquisition-wait-processing*

~ <unsigned integer>((0-2147483647))<<1>>(Milliseconds)

Specifies the interval during lock acquisition wait processing when `interval` is specified for the `pd_dbbuff_lock_release_detect` operand.

An overview of the lock acquisition processing for the global buffer is provided as follows. For this operand, specify the time in step 2.

1. If the lock for the global buffer is released, the lock is acquired. If the lock is successfully acquired, the process is terminated. However, if the lock cannot be acquired, the process is repeated up to the number of times specified for the `pd_dbbuff_lock_spn_count` operand.
2. If acquisition fails in step 1, the system sleeps for the amount of time

specified in this operand (waits for the specified amount of time to elapse).

3. Returns to step 1.

Condition

`interval` must be specified for the `pd_dbbuff_lock_release_detect` operand.

Specification guidelines

Determine the value for this operand by referring to *Lock release contention wait generation rate in global buffer lock release processing* in the *HiRDB Version 8 System Operation Guide*.

118) pd_dbbuff_wait_interval = *global-buffer-occupation-state-check-interval*

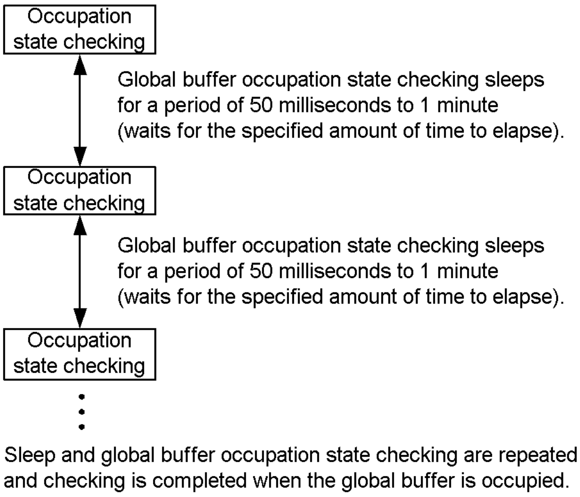
~ <unsigned integer>((0-2147483647))(milliseconds)

This operand applies only to a HiRDB/Parallel Server.

Specifies the interval at which the global buffer occupation state is to be checked. Specifying this operand changes the method of checking the global buffer occupation state.

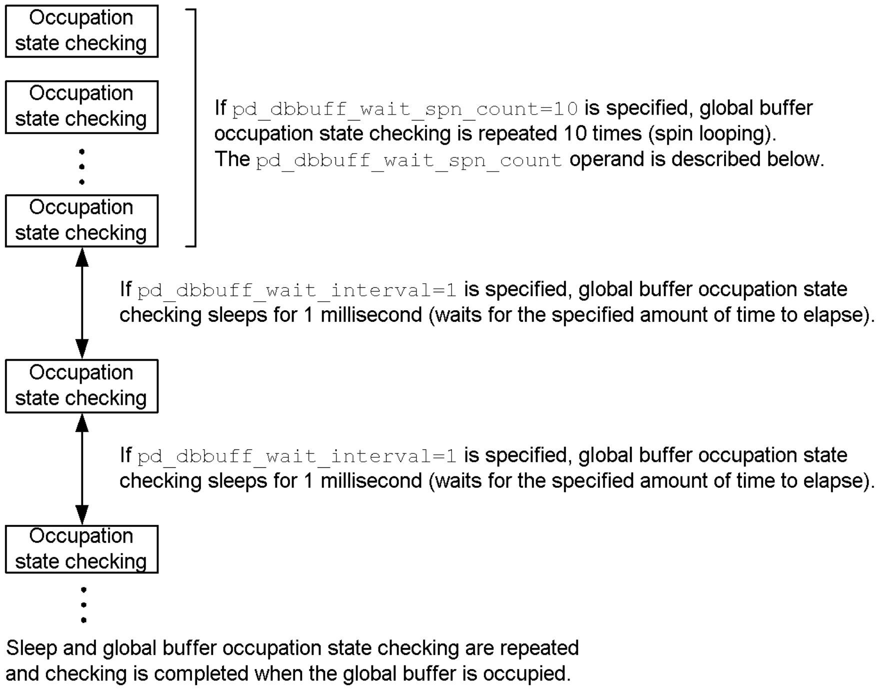
When this operand is not specified

The following processing occurs:


Occupation state checking
Global buffer occupation state checking sleeps for a period of 50 milliseconds to 1 minute (waits for the specified amount of time to elapse).
Occupation state checking
Global buffer occupation state checking sleeps for a period of 50 milliseconds to 1 minute (waits for the specified amount of time to elapse).
Occupation state checking
Sleep and global buffer occupation state checking are repeated and checking is completed when the global buffer is occupied.

When this operand is specified

The following processing occurs:


Occupation state checking
Occupation state checking
Occupation state checking
If pd_dbbuff_wait_spn_count=10 is specified, global buffer occupation state checking is repeated 10 times (spin looping).
The pd_dbbuff_wait_spn_count operand is described below.
If pd_dbbuff_wait_interval=1 is specified, global buffer occupation state checking sleeps for 1 millisecond (waits for the specified amount of time to elapse).
Occupation state checking
If pd_dbbuff_wait_interval=1 is specified, global buffer occupation state checking sleeps for 1 millisecond (waits for the specified amount of time to elapse).
Occupation state checking
Sleep and global buffer occupation state checking are repeated and checking is completed when the global buffer is occupied.

Specification guidelines

Specify this operand when all of the following conditions are satisfied. Performance may improve. Typically when this operand is used, a value of 1 is specified.

- Global buffer lock-release wait has occurred. (You can check for this based on `WAITL` in the execution result of the `pdbufls` command.)
- You want to improve performance even if the CPU usage rate must be increased.

If the CPU usage has become too high because 1 was specified in this operand, increase the value. If there is unused capacity in the CPU usage rate when 1 is specified in this operand, increase the `pd_dbbuff_wait_spn_count` operand value. Performance may improve.

119) pd_dbbuff_wait_spn_count = ***maximum-spin-loop-count-for-global-buffer-occupation-state-checking***

~ <unsigned integer>((0-2147483646))<<0>>

This operand is applicable to a HiRDB/Parallel Server.

Specifies the maximum spin loop count in an interval loop that can occur during global buffer occupation state checking. For details, see the description of the `pd_dbbuff_wait_interval` operand.

Specification guidelines

Normally, there is no need to specify this operand. Specify this operand when you specify 1 in the `pd_dbbuff_wait_interval` operand.

120) pd_dbbuff_rate_updpage = *deferred-write-trigger-request-rate*

～ <unsigned integer>((1-100))(%)

When you use the facility for parallel writes in deferred write processing, use this operand to specify the deferred write trigger as a percentage of the updated buffers. For details about the facility for parallel writes in deferred write processing, see the *HiRDB Version 8 Installation and Design Guide*.

Specification guidelines

Normally omit this operand. Deferred write processing may not always be completed within the synchronization point dump acquisition interval. You can specify this operand for such cases if you want to shorten the writing time by reducing the number of updated buffers and slightly reducing the updated buffer hit rate. A guideline for this operand's value is to use 50% (the initial value set by HiRDB) or determine the operand's value by referring to *Tuning deferred write processing* and *Tuning the synchronization point processing time when deferred write processing is used* in the *HiRDB Version 8 System Operation Guide*.

Relationships to other operands

This operand has the following relationships with the `pdbuffer` operand's `-y` option:

- The value set for the `pd_dbbuff_rate_updpage` operand applies to all the global buffers.
- The value of the `pdbuffer` operand's `-y` option applies to each global buffer.
- The `pdbuffer` operand's `-y` option takes precedence over the `pd_dbbuff_rate_updpage` operand.
- If the `pdbuffer` operand's `-y` option is omitted, the number of update buffer sectors for deferred write trigger event depends on the specification of this operand, as shown below:

pd_dbbuff_rate_updpage operand value	Number of update buffer sectors for deferred write trigger event
Specified	Number of global buffer sectors x `pd_dbbuff_rate_updpage` operand value

pd_dbbuff_rate_updpage operand value	Number of update buffer sectors for deferred write trigger event
Omitted	Determined automatically by HiRDB

2.3.23 Operands related to table or index reservation count

121) pd_assurance_table_no = *minimum-guaranteed-tables-count*

~ <unsigned integer>((0-4194294500))<<500>>

If you use the free space reusage facility, consider whether to specify this operand. For details about the free space reusage facility, see the *HiRDB Version 8 Installation and Design Guide*. Specifies the minimum guaranteed value for the table reservation count.

If the current number of tables that have been defined is equal to or less than the table reservation count, the free space reusage facility can be used because the table management information can be made resident in memory. If the current number of tables that have been defined exceeds the table reservation count, the free space reusage facility can no longer be used. Note that, for a HiRDB/Parallel Server, the table reservation count is applied to each back-end server. For example, if the table reservation count is `500`, the table reservation count of each back-end server is 500.

Specification guidelines

- When the `pd_sysdef_default_option` operand is omitted or when `recommendable` is specified in the `pd_sysdef_default_option` operand

  current-number-of-tables-using-free-space-reusage-facility + *number-of-tables-using-free-space-reusage-facility-to-be-defined-between-next-HiRDB-startup-and-termination*

- When `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand

  When HiRDB is started normally, the table reservation count automatically increases according to the number of tables for which the free space reusage facility is specified. Therefore, normally you need not specify this operand. However, if you add 101 or more tables that use the free space reusage facility between the startup and termination of HiRDB, specify the larger of the following two values as the table reservation count for this operand.

1. *current-number-of-tables-using-free-space-reusage-facility* + *number-of-tables-using-free-space-reusage-facility-to-be-defined-between-next-HiRDB-startup-and-termination*

2. *current-number-of-tables-using-free-space-reusage-facility* + 100

Notes

- For a HiRDB/Parallel Server, compute the table reservation counts for the individual back-end servers and specify the largest among them for this operand.
- For a row partitioned table, count each partitioned table as one table.
- If you use the inner replica facility, count the tables stored in the replica RDAREA as separate tables.
- If the specified value is unnecessarily large, a shared memory shortage may prevent HiRDB (the back-end server for a HiRDB/Parallel Server) from starting. For the formula for computing the size of shared memory, see the *HiRDB Version 8 Installation and Design Guide*.
- When `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand and the `pd_bes_shmpool_size` or `pd_sds_shmpool_size` operand is omitted in order to use the facility for automatically computing the shared memory size, make sure that the value of this operand is at least equal to the number of tables using the free space reusage facility at system startup plus 100.

Relationship to other operands

This operand's default value depends on the value specified in the `pd_sysdef_default_option` operand, as shown below:

- `recommendable` (default value): 500
- `v6compatible`: 0
- `v7compatible`: 100

122) pd_assurance_index_no = *minimum-guaranteed-indexes-count*

~ <unsigned integer>((50-4194294500))<<500>>

Specifies the minimum guaranteed value for the index reservation count. When the index reservation count is equal to or less than this operand's value, the index management information can be made resident in memory; this results in the following benefits:

- Performance is improved because index information is made resident in memory
- Statistical information is collected for indexes
- Output of duplicated messages is suppressed

If the index reservation count exceeds this operand's value, these benefits may no longer be applicable.

In the case of a HiRDB/Parallel Server, note that this operand's value applies to each back-end server. For example, if this operand is set to `600`, the minimum guaranteed index reservation count is 600 for each back-end server.

Specification guidelines

Specify in this operand the value obtained from the following formula:

Current index reservation count + number of indexes to be added before the next HiRDB startup

The following describes how to calculate the index reservation count:

- Count 1 index reservation for each index.
- In the case of a row-partitioned index, count 1 index reservation for each partition (for example, 3 partitions results in an index reservation count of 3).
- If you use the inner replica facility, count the indexes stored in the replica RDAREA as separate indexes.
- For a HiRDB/Parallel Server, compute the index reservation counts for the individual back-end servers and specify the largest among them for this operand.

If `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand, the index reservation count is obtained automatically by HiRDB, so normally you will not need to specify this operand in such a case. However, if the number of indexes you define exceeds 1.2 times the number of indexes in existence at the time of HiRDB startup, then at some point between startup and termination of HiRDB you must specify this operand in accordance with the guidelines provided above. For a HiRDB/Single Server, the index reservation count for the data dictionary table (124) must also be included, so you must add 124 to your obtained value.

Notes

- If the specified value is unnecessarily large, a shared memory shortage may prevent HiRDB (the back-end server for a HiRDB/Parallel Server) from starting.

  For the formula for computing the size of shared memory, see the *HiRDB Version 8 Installation and Design Guide*.
- When `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand and the

pd_bes_shmpool_size or pd_sds_shmpool_size operand is omitted in order to use the facility for automatically computing the shared memory size, make sure that the value of this operand is greater than 1.2 times the number of indexes in existence at the time of system startup.

Relationship to other operands

This operand's default value depends on the value specified in the pd_sysdef_default_option operand, as shown below:

HiRDB server type	pd_sysdef_default_option operand value	Default value
HiRDB/Single Server	recommendable (default value)	500
	v6compatible or v7compatible	Number of indexes in data dictionary table + 50
HiRDB/Parallel Server	recommendable (default value)	500
	v6compatible or v7compatible	50

- If you use the facility for automatically computing the shared memory size by omitting the pd_bes_shmpool_size operand in the server common definition, you must specify for the pd_assurance_index_no operand a value that is larger than 1.2 times the index count at the system startup.

2.3.24 Operands related to referential and check constraints

123) pd_constraint_name = LEADING | TRAILING

Specifies whether to specify a constraint name definition before or after constraint definition in referential constraint or check constraint.

LEADING: Specifies a constraint name definition before constraint definition (standard SQL specification).

TRAILING: Specifies a constraint name definition after constraint definition (XDM/RD-compatible specification).

124) pd_check_pending = USE | NOUSE

Specifies whether or not the check pending status (state that prevents the use of data for which integrity can no longer be guaranteed) is to be used in referential constraint or check constraint.

USE:

Use the check pending state.

NOUSE:

Do not use the check pending state.

Specification guidelines

Normally specify USE. If NOUSE is specified in this operand, data integrity may not be guaranteed. If processing performance is more important than data integrity, then specify NOUSE.

2.3.25 Operands related to HiRDB file system areas

125) pd_large_file_use = Y | N

Specifies whether or not large files can be used. For details on large files, see the *HiRDB Version 8 Installation and Design Guide*.

Y: Use large files.

N: Do not use large files.

Note

Be careful when changing the specification of this operand from Y to N. When this change is made, the RDAREAs inside the HiRDB file system area that has been created as a large file go into the error shutdown state. To free these RDAREAs from error shutdown, you need to use the pdfmkfs command to re-create the HiRDB file system area.

2.3.26 Operands related to the facility for predicting reorganization time

For details about how to use the facility for predicting reorganization time, see the *HiRDB Version 8 System Operation Guide* and *HiRDB Version 8 Command Reference*.

126) pd_rorg_predict = Y | N

Specifies whether to use the facility for predicting reorganization time.

Y: Uses the facility for predicting reorganization time.

N: Does not use the facility for predicting reorganization time.

Notes

- To execute the facility for predicting reorganization time, you must create a data dictionary RDAREA for storing the analysis information table and the operation history table.
- If you specify Y for this operand, shared memory is used. For details about the size of the shared memory to be used, see the *HiRDB Version 8 Installation and Design Guide*.

2.3.27 Operands related to security audit facility

For the method of using the security audit facility, see the *HiRDB Version 8 System Operation Guide*.

127) pd_audit = Y | N

Specifies whether to begin collecting an audit trail when HiRDB (a unit for a HiRDB/Parallel Server) is started.

Y: Begins collecting an audit trail when HiRDB is started.

N: Does not begin collecting an audit trail when HiRDB is started.

Even if Y is specified for this operand, you can still collect an audit trail by executing the pdaudbegin command.

Conditions

All of the following conditions must be satisfied. If Y is specified when all of these conditions are not satisfied, HiRDB (a unit for a HiRDB/Parallel Server) cannot be started.

- A HiRDB file system area has been created for an audit trail file.
- The name of the HiRDB file system for the audit trail file is specified for the pd_aud_file_name operand.

128) pd_aud_file_name = *HiRDB-file-system-area-name-for-audit-trail-file*

~ <pathname>((maximum of 150 characters))

This operand is required if you use the security audit facility. If you do not specify this operand, you cannot use the security audit facility.

Specify an absolute path name for the name of the HiRDB file system area for an audit trail file.

Notes

- When this operand is specified, HiRDB (a unit for a HiRDB/Parallel Server) cannot be started if an error occurs during the access to the HiRDB file system area for audit trail files.
- In the case of a system configuration in which multiple units run on the same server machine, use this operand to specify an audit trail file for each unit. If the same audit trail file is specified in pd_aud_file_name operands in the system common definition for multiple units on the same server machine, the correct audit trail cannot be acquired. However, for a unit to which the standby-less system switchover (effects distributed) facility is applied, specify this operand in the system common definition.

129) pd_aud_max_generation_size = *audit-trail-file-maximum-size*

~ <unsigned integer>((1-5240))<<100>> (MB)

Specifies the maximum size (MB) of audit trail files.

Specification guidelines

Because HiRDB needs 20 MB for management, determine the value for this operand so that the following condition is satisfied:

pd_aud_max_generation_size-value ×
pd_aud_max_generation_num-value <
size-of-HiRDB-file-system-area-for-audit-trail-files (value of the `-n` option of the `pdfmkfs` command) - 20 MB

130) pd_aud_max_generation_num = *maximum-audit-trail-file-count*

~ <unsigned integer>((2-200))<<50>>

Specifies the maximum number of (number of generations of) audit trail files to be created inside the HiRDB file system area for audit trail files.

Specification guidelines

- Hitachi recommends that you not specify the maximum value (200) in case errors occur in all audit trail files. For the method of handling errors in audit trail files, see the *HiRDB Version 8 System Operation Guide*.
- Because HiRDB needs 20 MB for management, determine the value for this operand so that the following condition is satisfied:

pd_aud_max_generation_size-value ×
pd_aud_max_generation_num-value <
size-of-HiRDB-file-system-area-for-audit-trail-files (value of the `-n` option of the `pdfmkfs` command) - 20 MB

Notes

During the startup of HiRDB (a unit for a HiRDB/Parallel Server), if there is a file with a generation number that is greater than the value specified for this operand, the specified value becomes invalid. In this case, the largest generation number is assumed as the maximum number of audit trail files to be created inside the HiRDB file system area.

131) pd_aud_no_standby_file_opr = down | <u>forcewrite</u>

Specifies the process to be performed by HiRDB when no swappable audit trail file is available.

`down`:

When the number of remaining swappable audit trail files is reduced to one, HiRDB (a unit for a HiRDB/Parallel Server) is forcibly terminated. After restarting HiRDB, load the data waiting to be loaded from the audit trail files into an audit trail table. During this step, temporarily suppress the execution of commands and transactions.

`forcewrite`:

If no swappable audit trail file is available, an audit trail file waiting for data loading (except for files that are shut down) is forcibly made into a swapping target, and audit trail outputting is continued. For this process, the audit trail file waiting for data loading that has the oldest update date is made into the swapping target.

If all files are shut down, audit trail outputting is cancelled.

132) pd_aud_async_buff_size = *size-of-buffer-used-for-asynchronous-output-of-audit-trail-file*

~ <unsigned integer>((0, 4096-6553600))<<401408>> (Byte)

Specifies the size (bytes) of the buffer to be used for asynchronously outputting the audit trail. If `0` is specified, the audit trail is synchronously output. The following table describes the advantages and disadvantages of each output method.

pd_aud_async_buff_size value	Audit trail output method	Advantages	Disadvantages
`0`	Synchronous output	Audit trail can be reliably output to an audit trail file.	Because file input/out occurs on the extension of SQL processing, the impact on performance is large.
`4096-6553600`	Asynchronous output	Can reduce the impact on SQL processing performance.	If HiRDB (a unit for a HiRDB/ Parallel Server) is abnormally terminated after the audit trail is output to the buffer and before it is output to an audit trail file, the audit trail may be lost.

Operand rule

For this operand, specify an integer multiple of 4096. If a value that is not an integer multiple of 4096 is specified, it is rounded up to an integer multiple of 4096 and set as the value for this operand. For example, if `5000` is specified, `8192` is set for the operand.

Relationship to other operands

If `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand, the default value for this operand is `4096`.

133) pd_aud_async_buff_count = ***number-of-buffer-sectors-used-for-asynchronous-output-of-audit-trail-file***

～ <unsigned integer>((1-6500))<<max(1, number of HiRDB servers in unit x 10)>> (sectors)

Specifies the number of buffer sectors to be used for asynchronously outputting an audit trail.

Relationship to other operands

If `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand, the default value for this operand is `3`.

134) pd_aud_async_buff_retry_intvl = ***retry-interval-for-allocation-of-a-buffer-to-be-used-for-asynchronous-output-of-audit-trail-file***

～ <unsigned integer>((1-1000))<<50>> (milliseconds)

Specifies the retry interval for monitoring for a buffer to be used for asynchronous output of the audit trail so that the audit trail can be acquired when all buffers are in use.

Specification guidelines

Normally, there is no need to specify this operand.

When the security audit facility is used and a UAP requires an extended amount of time to execute, specifying a small value in this operand may reduce the UAP execution time.

135) pd_aud_file_wrn_pnt = ***warning-message-output-trigger*****[,*****trigger-for-resetting-warning-message-output-status*****]**

warning-message-output-trigger**: ～ <unsigned integer><<0-100>><<0 or 80>>(%)**

When the number of unswappable audit trail files reaches or exceeds the warning value, a warning message is issued. For this operand, specify the warning value as a percentage of the maximum audit trail file count specified in the `pd_aud_max_generation_num` operand. For example, if `100` is specified for the `pd_aud_max_generation_num` operand, and `90` is specified for the `pd_aud_file_wrn_pnt` operand, the `KFPS05123-W` warning message is issued when the number of unswappable audit trail files reaches or exceeds 90.

For a HiRDB/Parallel Server, the number is checked for each unit.

If `0` is specified in this operand, no warning message is issued.

Relationship to other operands

- If this operand is omitted, and if `MANUAL` is specified for the `pd_watch_resource` operand or this operand is omitted, `0` is assumed for the `pd_aud_file_wrn_pnt` operand. That is, no warning message is issued.
- If this operand is omitted, and if `AUTO` is specified for the `pd_watch_resource` operand or this operand is omitted, `80` is assumed for the `pd_aud_file_wrn_pnt` operand. That is, a warning message is issued when 80% is reached or exceeded.

***trigger-for-resetting-warning-message-output-status*: ~ <unsigned integer><<0-99>>(%)**

Specifies the trigger for resetting the warning message output status. When the warning message (`KFPS05123-W`) is output, HiRDB goes into the warning message output status. Once HiRDB goes into this status, the warning message is not output again even if the number of unswappable audit trail files exceeds the warning value again. However, when the number of unswappable audit trail files falls below the trigger for resetting the warning message output status specified here, the warning message output status is reset.

For example, if `pd_aud_file_wrn_pnt=90,70` is specified, the warning message is output when the number of unswappable audit trail files reaches or exceeds 90% of the maximum number of audit trail files. Afterwards, no warning message is output until the number of unswappable audit trail files falls below 70% of the maximum number of audit trail files. After the percentage falls below 70%, and when it subsequently reaches or exceeds 90% again, the warning message is output.

Notes

- When this specification is omitted, warning-message-output-trigger -`30` is assumed as the default (if the result is a negative number, `0` is used).
- If a value greater than the warning message output trigger is specified, the warning message output trigger value is used.

2.3.28 Operands related to the system switchover facility

136) pd_ha = use | <u>nouse</u>

Specifies whether or not the system switchover facility is to be used.

`use`: Use the system switchover facility.

`nouse`: Do not use the system switchover facility.

Notes

When Real Time SAN Replication based on the log-only synchronous method is used, the system switchover facility cannot be used at the log application site. When `use` is specified in this operand, the `KFPS04689-W` message is displayed.

137) pd_ha_ipaddr_inherit = Y | N

Specifies whether or not IP addresses are to be inherited when the system switchover facility is used. How the system switchover facility operates depends on whether or not IP addresses will be inherited; for details, see the *HiRDB Version 8 System Operation Guide*.

`Y`: Inherit IP addresses.

`N`: Do not inherit IP addresses.

If you use the standby-less system switchover facility, omit this operand.

Specification guidelines

If you use the rapid system switchover facility, use the following guidelines when specifying a value:

- HiRDB/Single Server

  Because IP addresses cannot be inherited when you use the rapid system switchover facility, specify `N`.

- HiRDB/Parallel Server

  To inherit IP addresses, specify `Y`; otherwise, specify `N`. Note that the units that are the targets of the rapid system switchover facility cannot inherit IP addresses. Therefore, if you specify `Y` here, you need to specify `N` for the `pd_ha_ipaddr_inherit` operand of the unit control information definition of the units that are the targets of the rapid system switchover facility.

  If you specify `N` for this operand, you cannot specify `Y` for the `pd_ha_ipaddr_inherit` operand of the unit control information definition.

138) pd_ha_switch_timeout = Y | N

This operand can be specified when the server mode is used. Specification of this operand is invalid in the monitor mode.

This operand specifies whether to switch the system without waiting for internal termination processing of HiRDB when internal termination processing of HiRDB (a unit for a HiRDB/Parallel Server) during system switchover has exceeded the server failure monitoring time. The server failure monitoring time referred to here is the time specified for the `patrol` operand of the HA monitor or Hitachi HA Toolkit Extension.

For details about the `patrol` operand of the Hitachi HA Toolkit Extension, see the manual *Hitachi HA Toolkit*.

`Y`:

Switches the system without waiting for internal termination processing of HiRDB when internal termination processing of HiRDB during system switchover has exceeded the server failure monitoring time. In this case, system switchover is carried out by assuming that HiRDB has slowed down.

If you are using the standby-less system switchover (1:1) facility or the standby-less system switchover (effects distributed) facility, the specification for this operand is invalid during planned system switchover.

`N`:

Does not switch the system until internal termination processing of HiRDB during system switchover is terminated.

Advantage

If internal termination processing of HiRDB during system switchover takes a long time because, for example, of a disk error system switchover may be delayed as a result. If you specify `Y` (default value) for this operand, you can switch the system without waiting for internal termination processing of HiRDB, even when it is taking a long time.

Notes

- If you specify `Y` for this operand when the `patrol` operand value is small, planned system switchover may turn into system switchover based on slow-down. This is because internal termination processing of HiRDB during planned system switchover exceeds the time specified by the `patrol` operand.
- You need to be careful if `restart` is specified for the `switchtype` operand of the HA monitor or Hitachi HA Toolkit Extension. If `pd_ha_switch_timeout=Y` (default value) is specified, and if internal termination processing of HiRDB exceeds the server failure monitoring time, HiRDB is not started in the system in which the failure occurred. In this case, the system is immediately switched.

  For details about the `switchtype` operand of the Hitachi HA Toolkit Extension, see the manual *Hitachi HA Toolkit*.

139) pd_ha_prc_cleanup_check = Y | <u>N</u>

When the server-mode system switchover facility is used, this operand specifies whether system switchover processing is to be placed on hold until all server processes have terminated at the target unit or user server. By specifying `Y` in this operand, you can guarantee that all server processes that access the shared disk will have terminated before system switchover is performed in response to a server failure.

Note that the switchover target is a unit in the case of the standby system switchover facility or the standby-less system switchover (1:1) facility, and a back-end server in the case of the standby-less system switchover (effects distributed) facility.

`Y`: Perform system switchover only after all processes have terminated

`N`: Do not wait for termination of target processes

Before you specify this operand, see *Shared disk switchover control* in the *HiRDB Version 8 System Operation Guide*.

Conditions

All the following conditions must be satisfied:

- `use` is used in the `pd_ha` operand
- `server` is specified in the `pd_ha_acttype` operand
- The `pd_ha_unit` operand is omitted

Additionally, if you specify `Y` in this operand, use an HA monitor whose version is 01-08 or later.

140) pd_ha_mgr_rerun = <u>wait</u> | notwait

This operand applies only to a HiRDB/Parallel Server.

Specifies whether to wait for the completion of start processing of other units before switching the system for the system manager unit.

`wait`:

Waits for the completion of start processing of other units before switching the system for the system manager unit. In this case, the following checks are performed to start the system manager unit:

- Version check for each unit
- Is the dictionary server running?
- Is at least one front-end server running?
- Is at least one back-end server running?

If another unit has stopped, system switchover for the system manager unit may take a long time or fail. The following table describes what happens when system switchover for the system manager unit occurs when some of the units are stopped.

Reduced activation specification (pd_start_level value)	Specification of a name for the unit that does not start	Action
Not specified (0)	—	Cannot start.
Specified (1)	Nothing is specified for the `pd_start_skip_unit` operand	System switchover occurs after a wait time specified by the `pd_reduced_check_time` operand.*
	Stopped unit is specified for the `pd_start_skip_unit` operand.	System switchover is completed immediately.*

Legend:

— : Not applicable

* System switchover is completed only when all of the following conditions are satisfied:

- The dictionary server is running.
- At least one front-end server is running.
- At least one back-end server is running.

`notwait`:

Does not wait for the completion of start processing of other units before switching the system for the system manager unit. When this operand is specified, system switchover is executed rapidly for the system manager unit without waiting for other units that have stopped to start.

When system switchover is completed for the system manager unit, the message `KFPS05210-I` (system initialization completion message) is issued even if the HiRDB operating environment is not ready. If a job cannot be continued even after the completion of the system manager switchover because of a UAP error, for example, use the `pdls` command to check the system operating status.

Condition

To specify `notwait` for this operand, the system configuration must satisfy a certain condition. For details about the required system configuration, see the *HiRDB Version 8 System Operation Guide*.

Specification guidelines

To prevent failure of the system switchover for the system manager unit, specify `notwait`. For details about how to handle system manager unit switchover failures, see the *HiRDB Version 8 System Operation Guide*.

141) pd_ha_transaction = <u>error</u> | queuing

Specifies whether to use the transaction queuing facility. Also specifies the processing that takes place when the number of connections to the HiRDB server exceeds the maximum number of concurrent connections (value specified by the `pd_max_users` operand) during system switchover. For details on for the transaction queuing facility, see the *HiRDB Version 8 System Operation Guide*.

`error`:

- Does not use the transaction queuing facility. The transactions being processed by the back-end server or dictionary server being switched end in errors.
- If the number of connections to the HiRDB server exceeds the maximum number of concurrent connections during system switchover, the connections to the HiRDB server end in errors.

`queuing`:

- Uses the transaction queuing facility. Instead of ending in errors, the transactions being processed by the back-end server or dictionary server being switched are queued by the front-end server until system switchover is completed.
- If the number of connections to the HiRDB server exceeds the maximum number of concurrent connections during system switchover, connection processing to the HiRDB server is retried for the duration of `pd_ha_trn_queuing_wait_time` + `pd_ha_trn_restart_retry_time`. However, the HiRDB client version must be 07-00 or newer.

Conditions

If you use the transaction queuing facility, all of the following conditions must be satisfied:

- HiRDB/Parallel Server
- The rapid system switchover facility, standby-less system switchover (1:1) facility, or standby-less system switchover (effects distributed) facility is used.

Note that these conditions need not be satisfied if connection processing to the HiRDB server is to be retried by a HiRDB client.

Relationship to client environment definition

Even when `queuing` is specified for this operand, you can specify that the transaction queuing facility not be used for each client. To cancel the transaction queuing facility for each client, specify the `PDHATRNQUEUING` operand in the client environment definition. For details about the `PDHATRNQUEUING` operand, see the *HiRDB Version 8 UAP Development Guide*.

Relationship to other operands

This operand is related to the following operands:

- `pd_ha_trn_queuing_wait_time`
- `pd_ha_trn_restart_retry_time`

142) pd_ha_trn_queuing_wait_time = *transaction-queuing-wait-time*

~ <unsigned integer>((1-3600))<<180>>(Seconds)

Specifies the transaction queuing wait time when the transaction queuing facility is used. If the standby unit or server does not start within the wait time specified by this operand, the transactions being queued end in errors. Transactions that occur subsequently end in errors without being queued. If a unit or server starts before this wait time, transactions are resumed without waiting for the time specified by this operand.

If the standby-less system switchover (1:1) facility is used, the alternate portion becomes the standby system during normal operation, and the normal BES unit becomes a standby system during alternation.

Condition

The `pd_ha_transaction` operand must be set to `queuing`.

Specification guidelines

Normally, you need not specify this operand. However, if the rollforward during system switchover takes 180 seconds or longer, increase the value of this operand.

143) pd_ha_trn_restart_retry_time = *retry-time-upper-limit-after-transaction-start-request-errors*

~ <unsigned integer>((1-3600))<<60>>(Seconds)

If system switchover occurs while the transaction queuing facility is being used, transactions are queued by the front-end server. However, during the period between the system switchover and the restart of the standby unit or server, the front-end server cannot detect the system switchover. During this period (the period between the system switchover and the restart of the standby unit or server), the front-end server issues a transaction start request to the running unit or server. However, because the running unit has already been abnormally terminated, this transaction start request ends in an error. For the transaction that ends in an error, the front-end server re-issues a transaction start request (retries the transaction start request).

This operand specifies the upper limit for this retry time. If the standby unit or server is not restarted within the value specified by this operand, the transactions being retried end in errors. Furthermore, transactions that occur subsequently end in errors without being retried. Note that if the standby unit or server begins to restart before this retry time, no retries occur and transactions are queued.

Condition

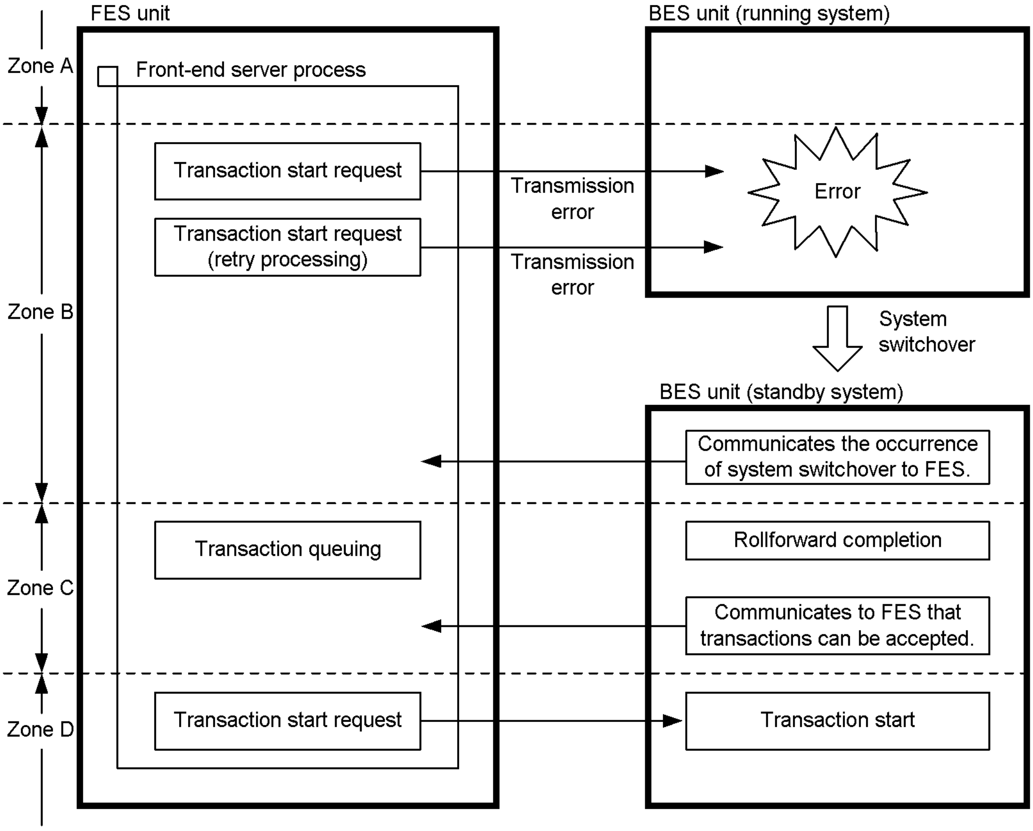
The `pd_ha_transaction` operand must be set to `queuing`.

Specification guidelines

Normally, you need not specify this operand. However, if disk switching by cluster software takes 60 seconds or longer, increase the value of this operand.

Remarks

The relationship between the `pd_ha_trn_queuing_wait_time` and `pd_ha_trn_restart_retry_time` operands is explained as follows.


FES unit
BES unit (running system)
Zone A
Front-end server process
Transaction start request
Transmission error
Error
Transaction start request (retry processing)
Transmission error
Zone B
System switchover
BES unit (standby system)
Communicates the occurrence of system switchover to FES.
Transaction queuing
Rollforward completion
Zone C
Communicates to FES that transactions can be accepted.
Zone D
Transaction start request
Transaction start

Explanation:

Zones A and D:

Transactions can be started (normal state).

Zone B:

The unit containing the back-end server is being switched, and the front-end server cannot detect this condition. Transaction start request is retried for the duration of time specified by the `pd_ha_trn_restart_retry_time` operand. When the front-end server detects the system switchover, transactions are queued. If the front-end server cannot detect it within the specified time, the transactions end in errors.

Zone C:

The unit containing the back-end server is being switched, and the front-end server has detected this condition. Transactions are queued for the duration of time specified by the `pd_ha_trn_queuing_wait_time` operand. If the transactions cannot be started within the specified time, they end in errors.

144) pd_ha_resource_act_wait_time = *maximum-wait-time-for-resource-activation*]

~ <unsigned integer>((2-3600))<<10>> (seconds)

When you use the standby-less system switchover (effects distributed) facility, this operand specifies the maximum wait time for the running server's resources to be activated at the time of unit startup. Unit startup processing is placed on hold up to the specified amount of wait time. If the resources are activated within the specified amount of time, unit startup processing resumes.

Advantages

When unit startup processing is completed, jobs can be started only if the running server's startup processing is completed in the unit.[#] By specifying an appropriate value in this operand, you can start your jobs immediately after the unit's startup processing is completed because the unit startup processing will have waited on wait status for resource activation.

#: This event may occur if it takes time to process the command specified in the `usrcommand` operand in the HA monitor's `sysdef` file.

Specification guidelines

Normally, there is no need to specify this operand. You should specify this operand when all the following conditions are applicable:

- The standby-less system switchover (effects distributed) facility is being used
- The `KFPS05623-I` message was displayed
- The target unit for the message contains the running server

Use the following guideline to determine the value to be specified in this operand:

10 + time required for resource activation processing (seconds)

Time required for resource activation processing: Maximum processing time for the command specified in the `usrcommand` operand in the HA monitor's `sysdef` file

Remarks

If the unit does not contain the running server, the running server's startup is placed on hold for the amount of time specified in this operand. However, if all the servers in the unit start as standby servers, unit startup processing is restarted without waiting for the amount of time specified in this operand.

2.3.29 Operands related to HiRDB Datareplicator

145) pd_rpl_init_start = Y | N

Specifies whether or not the HiRDB Datareplicator linkage facility is to be used from the time of HiRDB startup.

Y: Use the HiRDB Datareplicator linkage facility from the time of HiRDB startup.

N: Do not use the HiRDB Datareplicator linkage facility from the time of HiRDB startup.

Operation method

- When N is specified for this operand, the HiRDB Datareplicator linkage facility can be used by entering the pdrplstart command.
- The pdrplstop command is entered to stop the HiRDB Datareplicator linkage facility.
- When data linkage has been stopped by entering the pdrplstop command, HiRDB Datareplicator linkage can be restarted by entering the pdrplstart command. However, before the pdrplstart command is entered, the target database must be re-created based on the extracted database.

Notes

When Real Time SAN Replication based on the log-only synchronous method is used, the HiRDB Datareplicator linkage facility cannot be used at the log application site. If Y is specified in this operand, the KFPS04689-W message is displayed.

146) pd_rpl_reflect_mode = server | uap

This operand is applicable to a HiRDB/Parallel Server.

Specifies uap in order to use HiRDB Datareplicator's target transaction synchronization facility. If you do not use the target transaction synchronization facility, there is no need to specify this operand. For details about the target transaction synchronization facility, see the *HiRDB Datareplicator Version 8 Description, User's Guide and Operator's Guide*.

server: Incorporates transactions for each HiRDB server.

uap: Incorporates transactions for each UAP.

Condition

When you specify this operand, you must also specify the pd_rpl_hdepath operand. If the pd_rpl_hdepath operand is omitted, this operand is ignored.

Notes

When `uap` is specified, the amount of update log information increases. In such a case, refer to the *HiRDB Version 8 Installation and Design Guide* and re-estimate the total size of the system log files.

Relationship to other operands

If you specify `uap` in this operand, re-estimate the value of the `pd_log_max_data_size` operand.

147) pd_log_rpl_no_standby_file_opr = <u>stop</u> | continue

Specifies the action to be taken when a swap request is received when not all of the system log files can be created at the swapping destination because system log extraction by HiRDB Datareplicator using the HiRDB Datareplicator linkage facility has not been completed.

`stop`: Stop the HiRDB unit forcibly.

`continue`: Stop HiRDB Datareplicator linkage and continue operation with HiRDB only.

Condition

Specification of this operand is valid only if `Y` is specified for the `pd_rpl_init_start` operand or if the `pdrplstart` command has been entered and the HiRDB Datareplicator linkage facility is being used. In the case of a HiRDB/Parallel Server, this operand is not valid for front-end servers and dictionary servers.

Notes

- When `stop` (the default value) is specified for this operand and the HiRDB unit is terminated forcibly, the HiRDB unit must be restarted after it has been confirmed that the log file being extracted on the HiRDB Datareplicator side has moved to another file (i.e., at least one file has been extracted completely). Even if the HiRDB unit is restarted immediately after the forced termination, HiRDB will recognize the abnormal status and forcibly terminate the HiRDB unit again if there is no swap target file.
- When `continue` is specified for this operand and HiRDB Datareplicator linkage is stopped, a mismatch will occur between the extracted database and the target database on the HiRDB Datareplicator side. Therefore, it is necessary to re-create the target database.

2.3.30 Operands related to linkage to JP1

148) pd_jp1_use = Y | <u>N</u>

Specifies whether or not events such as HiRDB startup and termination are to be registered into JP1. For details about HiRDB events that can be registered, see the *HiRDB Version 8 Installation and Design Guide*.

`Y`: Registers HiRDB events in JP1.

`N`: Does not register HiRDB events in JP1.

Note

- For the Solaris and AIX 5L versions, this operand is permitted only for HiRDB in the 32-bit mode.
- If `Y` is specified for this operand on an OS that does not support registration of events in JP1, the message `KFPS04605-W` is issued during HiRDB startup.

149) pd_jp1_event_level = 1 | 2

Specifies the HiRDB event types (basic or extended attribute) that are registered in JP1. For details about basic and extended attributes, see the *HiRDB Version 8 Installation and Design Guide*.

`1`: Registers only the basic attributes.

2: Registers both the basic and extended attributes.

Condition

This operand is valid when `Y` is specified for the `pd_jp1_use` operand.

Specification guidelines

- Extended attributes can be registered only when the HiRDB version is 07-02 or later.
- As shown in the following table, the value that can be specified differs depending on the combination between the OS and the JP1 product that registers events.

OS		When the linked product is JP1/Base (JP1/IM)		When the linked product is JP1/System Event Service	
		Operand specification value		Operand specification value	
		1	2	1	2
HP-UX	32-bit mode	Y		Y*	N
	64-bit mode	Y	N	Y*	N
	IPF	Y		—	

OS		When the linked product is JP1/Base (JP1/IM)		When the linked product is JP1/System Event Service	
		Operand specification value		Operand specification value	
		1	2	1	2
Solaris	32-bit mode	Y		Y*	N
AIX	32-bit mode	Y		Y*	N
Linux	32-bit mode	Y		—	
	IPF	Y		—	
	IM64T	N		N	

Legend:

Y: Can be specified.

N: Cannot be specified.

—: Not supported.

* If both JP1/Base and JP1/System Event Service are installed, events are registered in JP1/Base.

150) pd_jp1_event_msg_out = Y | N

Specifies whether to register message-outputting events in JP1.

`Y`: Registers message-outputting events.

`N`: Does not register message-outputting events.

Condition

This operand is valid when `Y` is specified for the `pd_jp1_use` operand.

Specification guidelines

Specify this operand if you want to reduce the number of registrations in JP1 when the volume of messages output by HiRDB is large.

2.3.31 Operands related to Directory Server linkage facility

151) pd_directory_server = sods

Specify this operand if you use the directory server linkage facility. When this facility is used, a directory server (Sun Java System Directory Server) centrally manages the system user information, including that of HiRDB, and performs authentication of users attempting to connect to HiRDB. For details on the directory server linkage facility, see the *HiRDB Version 8 System Operation Guide*.

sods:

Specify this option to link to the Sun Java System Directory Server.

Conditions

- The following table shows the required (applicable OS) for using a directory server linkage facility.

  HP-UX

  Solaris 8

  AIX 5L

- HiRDB must be in the 32-bit mode and the POSIX Library version.
- To use a directory server linkage facility, you need the HiRDB LDAP Option.

2.3.32 Operands related to Real Time SAN Replication

For details about Real Time SAN Replication, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide*.

152) pd_rise_use = Y | N

Specifies whether to use Real Time SAN Replication.

Y: Uses Real Time SAN Replication.

N: Does not use Real Time SAN Replication.

If you specify Y for this operand, HiRDB uses TrueCopy and RAID Manage of SANRISE to manage and control database files and system files.

Relationship to other operands

If you specify Y in this operand, you must also specify the pd_rise_pairvolume_combination operand. If the pd_rise_pairvolume_combination operand is omitted, HiRDB cannot be started.

153) pd_rise_pairvolume_combination = sync | async | hybrid | syssync

Selects a Real Time SAN Replication processing method.

sync: All synchronous method

async: All asynchronous method

hybrid: Hybrid method

syssync: Log-only synchronous method

Conditions

- Y must be specified in the pd_rise_use operand.
- HiRDB Disaster Recovery Light Edition is required in order to specify syssync. If you specify syssync without installing HiRDB Disaster Recovery Light Edition, HiRDB cannot be started.

Notes

If you specify syssync when using the inner replica facility, HiRDB cannot be started.

Relationship to other operands

- If you specify Y in the pd_rise_use operand, you must specify this operand. If this operand is omitted in such a case, HiRDB cannot be started.
- If you specify sync, hybrid, or syssync, make sure that you also specify the pd_rise_fence_level operand. If the pd_rise_fence_level operand is omitted, HiRDB cannot be started.

154) pd_rise_disaster_mode = <u>normal</u> | alone

Specify this operand if you select the hybrid method for the Real Time SAN Replication processing method. This operand specifies whether to maintain data matching by establishing synchronization between the main site and the remote site.

normal:

Maintains data matching by establishing synchronization between the main site and the remote site.

alone:

Does not establish synchronization between the main site and the remote site, and thus does not maintains data matching. Specify this option when a factor such as a natural disaster makes it impossible to update a remote site by copying data.

Conditions

To specify this operand, all of the following conditions must be satisfied:

- Y is specified for the pd_rise_use operand.
- hybrid is specified for the pd_rise_pairvolume_combination operand.

Specification guidelines

This operand is used when a factor such as a natural disaster makes it impossible to update a volume located at a remote site by copying data. Therefore, normally omit this operand or specify `normal`.

155) pd_rise_fence_level = data | never

Specifies the action to be taken if synchronous copying of data to a volume at the remote site (transferring all or some HiRDB files) fails due to an error when the all synchronous method, hybrid method, or log-only synchronous method has been selected as the Real Time SAN Replication processing method. For details about the actions to be taken in the event of a synchronous copy error, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*

`data`: Stops the updating of the volumes located at the main site.

`never`: Updates only the volumes located at the main site and continues operation.

Note

When all of the following conditions are satisfied, you must specify this operand. Otherwise, HiRDB start processing ends in an error.

- `Y` is specified for the `pd_rise_use` operand
- `Sync` or `hybrid` is specified for the `pd_rise_pairvolume_combination` operand.

Relationship to other operands

If you specify `sync`, `hybrid`, or `syssync` operand in the `pd_rise_pairvolume_combination` operand, you must also specify this operand. If this operand is omitted, HiRDB cannot be started.

2.3.33 Operands related to inner replica facility

To use the inner replica facility, you need the HiRDB Staticizer Option. For details on the inner replica facility, see the manual *Inner Replica Facility HiRDB Staticizer Option Version 8*.

156) pd_inner_replica_control = *inner-replica-maximum-group-count*

~ <unsigned integer>((1-4194296))<<0>>

Specifies the maximum number of inner replica groups to be used. For this operand, specify a value that is greater than the number of inner replica groups used during normal startup of HiRDB.

For a HiRDB/Parallel Server, specify the largest inner replica group count among those managed by back-end servers For example, specify at least `31` for this operand for the following back-end server configuration:

- Back-end server 1: has 10 inner replica groups.
- Back-end server 2: has 20 inner replica groups.
- Back-end server 3: has 30 inner replica groups.

Notes

- If the actual number of inner replica groups is greater than the value specified for this operand, HiRDB cannot be started.
- The number of inner replica groups to be defined cannot exceed the value specified for this operand.
- Increasing the specification value of this operand increases the shared memory used by a single server or back-end server. Therefore, do not specify an unnecessarily large value. If a shortage occurs in the shared memory, HiRDB may not be able to start. For details on the shared memory used by a single server and back-end servers, see the *HiRDB Version 8 Installation and Design Guide*.
- Real Time SAN Replication based on the log-only synchronous method cannot be used together with the inner replica facility. If an attempt is made to use them together, HiRDB cannot be started.

Relationship to other operands

When this operand is specified, `NONE` is assumed in the `pd_indexlock_mode` operand.

157) pd_lv_mirror_use = Y | N

Specifies whether to set the open attribute of the replica RDAREA to `SCHEDULE`.

`Y`:

Sets the open attribute of the replica RDAREA to `SCHEDULE`.

`N`:

Sets the open attribute of the replica RDAREA to that of the original RDAREA. The open attribute of the original RDAREA is specified by the following operands:

- `pd_rdarea_open_attribute_use`
- `pd_rdarea_open_attribute`

Specification guidelines

- If you use the inner replica facility under Logical Volume Manager (LVM), the open attribute of the replica RDAREA must be set to

SCHEDULE. Therefore, specify Y for this operand. For details, see *Notes on Operation in Logical Volume Management* in the manual *Inner Replica Facility HiRDB Staticizer Option Version 8*.

- If you do not use LVM, specify N for this operand.

Note

Note that specifying Y for this operand increases the shared memory used by a single server or back-end server. If a shortage occurs in the shared memory, HiRDB may not be able to start. For details on the shared memory used by a single server and back-end servers, see the *HiRDB Version 8 Installation and Design Guide*.

158) pd_max_reflect_process_count = *number-of-processes-to-be-allocated-for-reflection-processing*

~ <unsigned integer>((1-256))

Specify this operand if you plan to perform updatable online reorganization. If you omit this operand, you cannot perform updatable online reorganization.

For this operand, specify the number of processes to be used for reflection processing. HiRDB allocates the number of processes specified by this operand. For a HiRDB/Parallel Server, the value specified for this operand indicates the number of processes per front-end server.

Specification guidelines

- The value of this operand must satisfy the following condition. Otherwise, HiRDB cannot be started.

  pd_max_reflect_process_count-value + *pd_max_users-value* ≤ 2000 (maximum value of pd_max_users)

- For the number of processes to be allocated for reflection processing (formula for estimating the value for the pd_max_reflect_process_count operand), see the manual *Inner Replica Facility HiRDB Staticizer Option Version 8*.

159) pd_log_org_reflected_logpoint = keep | release

Specify this operand if you plan to perform updatable online reorganization.

This operand specifies whether to change the status of the system log file for which reflection processing has been completed.

keep:

Does not change the status of the system log file for which reflection processing has been completed. Even when reflection processing has been completed for all update logs inside the file, the system log file is kept in the overwriting-denied status for online reorganization.

`release`:

Changes the status of the system log file for which reflection processing has been completed. When reflection processing has been completed for all update logs inside the file, the system log file status is changed from the overwriting-denied status for online reorganization to the overwriting-permitted status for online reorganization.

Note that in the following cases, the system log file status is changed from the overwriting-denied status for online reorganization to the overwriting-permitted status for online reorganization regardless of the value specified for this operand.

- All reflection processes have been terminated.
- The `pdorbegin -u` command is executed.
- The `pdorend -u` command is executed.

Specification guidelines

The values specified for this operand and the `pd_log_org_no_standby_file_opr` operand determine how the overwriting-denied status for online reorganization of the system log file changes and how updatable online reorganization is performed. For the specification guidelines for this operand, the manual *Inner Replica Facility HiRDB Staticizer Option Version 8*.

Note

Following forcible termination, abnormal termination, or planned termination, you can change the specification value of this operand from `keep` (default value) to `release` only.

160) pd_log_org_no_standby_file_opr = stop | continue

Specify this operand if you plan to perform updatable online reorganization.

This operand specifies the processing for HiRDB when system log file swapping occurs while all system log files are in the overwriting-denied status for online reorganization.

`stop`:

Forcibly terminates HiRDB. For a HiRDB/Parallel Server, the applicable unit is forcibly terminated.

`continue`:

The overwriting-denied status for online reorganization is changed to the overwriting-permitted status for online reorganization, and processing continues. For a HiRDB/Parallel Server, this option applies to the system log files inside the server.

Specification guidelines

The values specified for this operand and the `pd_log_org_reflected_logpoint` operand determine how the overwriting-denied status for online reorganization of the system log file changes and how updatable online reorganization is performed. For the specification guidelines for this operand, the manual *Inner Replica Facility HiRDB Staticizer Option Version 8*.

Note

If you forcibly release the overwriting-denied status for online reorganization by specifying `continue` for this operand. System log files used after that time do not go into the overwriting-denied status for online reorganization. Consequently, if a system log file that has an update log that has not been reflected to the main-system RDAREA is overwritten, reflection processing may not be executed. In this case, HiRDB continues executing jobs using only the current sub-system RDAREAs. To return the main-system RDAREA to the current status or to re-execute updatable online reorganization, you need to restore the main-system RDAREA from the sub-system RDAREA in advance. For the procedure for restoring the main-system RDAREA, see the manual *Inner Replica Facility HiRDB Staticizer Option Version 8*.

2.3.34 Operands related to HiRDB External Data Access facility

For details on the HiRDB External Data Access facility, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

Note that some platforms on which HiRDB can be run do not support the HiRDB External Data Access facility. For details, see *HiRDB External Data Access* in the manual *HiRDB Version 8 Description*.

161) pd_max_foreign_server = ***foreign-server-maximum-count***

~ <unsigned integer>((1-512))<<1>>

Specifies the maximum number of foreign servers that can be connected.

Condition

HiRDB External Data Access is required.

Specification guidelines

Specifying an unnecessarily large value for this operand needlessly increases the size of the shared memory used. If the size of the shared memory used is large, HiRDB may not be able to start. Therefore, add the number of currently connected foreign servers to the number expected to be added in the future, and specify the resulting number for this operand.

2.3.35 Operands related to OLTP

162) pd_oltp_holdcr = use | nouse

Specifies whether to use the holdable cursor facility in a UAP under the OLTP environment.

`use`: Uses the holdable cursor facility in a UAP under the OLTP environment.

`nouse`: Does not use the holdable cursor facility in a UAP under the OLTP environment.

Specification guidelines

Specify `use` for this operand only when you plan to use the holdable cursor facility in a UAP under the OLTP environment.

Note

To use the holdable cursor facility in a UAP under the OLTP environment, you need to use a UAP that satisfies a certain condition. For details about the holdable cursor, see the *HiRDB Version 8 UAP Development Guide*.

2.3.36 Operands related to version upgrade

163) pd_auto_vrup = Y | N

Specifies whether or not the `pdvrup` command for executing HiRDB version upgrading is to be started automatically. For details about how to upgrade the HiRDB version, see the *HiRDB Version 8 Installation and Design Guide.*

`Y`: Start the command automatically.

`N`: Do not start the command automatically (the HiRDB administrator must enter the `pdvrup` command).

Specification guidelines

Specify `Y` to upgrade a large number of HiRDB systems by using JP1/ Software Distribution to distribute programs. This will eliminate the need to enter the `pdvrup` command at each HiRDB system.

164) pd_sysdef_default_option = recommendable | v6compatible | v7compatible

Specifies use of a previous version's default values. This operand's default value depends on the version.

`recommendable`: Applies the default values of the most recent version.

`v6compatible`: Applies the default values of HiRDB Version 6 or earlier.

`v7compatible`: Applies the default values of HiRDB Version 7.

Specification guidelines

Normally, omit this operand. This operand is used when problems arise from changing to the default values set by upgrading. For details about the applicable operands, see *1.4(1) Operands whose default value depends on the version*.

2.3.37 Operands related to communication processing

165) pd_service_port = *port-number-for-high-speed-connection-from-client*

～ <unsigned integer>((5001-65535))

Specifies the port number for high-speed connection. This operand is specified when the high-speed connection facility is used. The following rules apply to the port number that is specified:

- The port number must be unique within the host
- A port number that is different from the port number specified in `pd_name_port` must be specified

For details about the high-speed connection facility, see the *HiRDB Version 8 Installation and Design Guide.*

Operation method

Specify the port number specified with this operand in the client environment definition (`PDSERVICEPORT` operand) of the client to which the high-speed connection facility is to be applied.

If the mutual system switchover facility is used, a different port number must be specified for each HiRDB instance.

If multiple HiRDB systems or multiple HiRDB units will be started on the same machine, a different port number must be specified for each HiRDB instance.

If a firewall is installed in the HiRDB server, you need to fix the port number for client connection. By specifying this operand, you can fix the port number for client connection. The following client connection method is used in this case:

- If a firewall and NAT are installed in the HiRDB server, use a setting that uses the high-speed connection facility.
- Specify this operand if only a firewall is installed in the HiRDB server. There is no need to specify the client environment definition (`PDSERVICEPORT` operand).

For details about how to set the HiRDB environment when a firewall or NAT is installed in the HiRDB server, see the *HiRDB Version 8 Installation and*

Design Guide.

Notes

- If the port number specified in the pd_name_port operand is specified in pd_service_port, this specification is invalid, and the high-speed connection facility will not be used.
- If the pd_name_port operand is omitted, 20000 must not be specified in pd_service_port.
- Specify a number that is not within the range of port numbers assigned automatically by the OS. The range of port numbers assigned automatically by the OS depends on the OS.
- If a port number is specified that is within the range of port numbers assigned automatically by the OS, that number may be being used by another program. If so, the HiRDB server will not start.
- If a mutual system switchover configuration is used, specify this operand in the unit control information definition. Specify a different port number in the unit control information definition for each unit. If the same port numbers are specified, system switchover will fail in one of the units during a switching attempt.
- If the high-speed connection facility is used to concurrently issue a number of connection requests that exceeds the value specified for the pd_max_users operand, a shortage occurs in the number of active front-end servers and single servers for extracting connection requests from the message queue. Consequently, connection requests cannot be extracted from the message queue, and the message queue monitoring facility may stop the unit. Therefore, make sure that the number of concurrent connection requests generated does not exceed the value specified for the pd_max_users operand. For details about the message queue monitoring facility, see the *HiRDB Version 8 System Operation Guide*.

166) pd_change_clt_ipaddr = 0 | 1

Specifies the network to be used by the HiRDB server to communicate with HiRDB clients.Normally, this operand need not be specified.

0:

Communication from the HiRDB server to HiRDB clients uses the network in which the IP address specified in the PDCLTRCVADDR operand of the client environment definition is located. If the PDCLTRCVADDR operand is omitted, the IP address of the standard host is assumed.

1:

For communication from the HiRDB server to HiRDB clients, the network used for communication from HiRDB clients to the HiRDB server is used.

167) pd_registered_port =*"port-number-reservation-range"* [,*"port-number-reservation-range"*]...

~ **<character string>**

In the case of the Linux version, evaluate whether or not this operand should be specified.

Specifies ranges of port numbers to be used for communication by the HiRDB server.

HiRDB reserved port facility is valid only for server-to-server communication. This facility need not be used if the number of ports to be used is small.

Operand specification example

This example provides the following 35,000 port numbers: 6000 to 8999, 12500 to 29999, and 30500 to 44999:

```
set pd_registered_port =
"6000:8999","12500:29999","30500:44999"
```

Advantages

The port numbers to be used for communication between a HiRDB server and HiRDB clients or between HiRDB servers are assigned automatically by the OS. If the communication volume increases significantly, a shortage of port numbers may cause an interruption of processing or may adversely affect the communication processing by other programs. Using this operand, specify a range of port numbers to be used exclusively by HiRDB can prevent such problems.

Specification guidelines

The range of port numbers that can be specified is from 5001 through 49151.

For the number of ports used by HiRDB, see the *HiRDB Version 8 Installation and Design Guide*.

Specification value tuning method

As a guideline, the value obtained from the following formula is the total number of ports that will be needed:

a + b + 100

a: Number of HiRDB reserved ports that are being used (`# OF REGISTERED PORTS`)

b: Number of ports being used that were assigned automatically by the OS because all HiRDB reserved ports were in use (# OF ASSIGNED PORTS)

These values can be determined from the statistical information related to system operation obtained by executing the statistics analysis utility.

Operand rules

- Up to 10 ranges of port numbers can be specified.
- If more than one range is specified, ensure that there is no overlap in the port numbers included in the various ranges.
- An ending port number must be greater than the paired starting port number.

Notes

- For the HP-UX, Solaris, and AIX 5L versions, the range of port numbers assigned by the OS is extended automatically. Therefore, a shortage of port numbers will not occur for HiRDB even if this operand is omitted.
- Make sure that the port number reservation ranges specified in this operand do not overlap the port numbers listed below; if there is any overlap, a problem such as a HiRDB startup error may occur:
  - Port numbers specified by the pd_name_port operand
  - Port numbers specified by the pd_service_port operand
  - Port numbers specified by the -p option of the pdunit operand
  - Port numbers registered in /etc/services
  - The port number specified by the PDCLTRCVPORT operand in the HiRDB client environment definition (provided that the HiRDB exists on the same server machine).
  - Port numbers being used by other programs
  - Port numbers assigned automatically by the OS
- If multiple HiRDB servers reside on the same server machine (for example, in a multi-HiRDB structure or when the system switchover facility is used), ensure that the port number reservation ranges of the individual HiRDB servers do not overlap.
- This facility is valid only for server-to-server communication.
- Specify port numbers that are outside the range of port numbers assigned automatically by the OS. The range of port numbers assigned automatically by the OS depends on the OS.
- In the case of a port that does not support the HiRDB reserved port

facility, specification of its port number in this operand is ignored. For the scope of the HiRDB reserved port facility, see the `pd_registered_port_level` operand.

168) pd_registered_port_check = Y | N | C | W

Specifies whether or not checking is to be performed for overlapping port numbers in the ranges of port numbers specified in the `pd_registered_port` operand and in the port numbers registered in

`/etc/services` (definition location in an NIS or DNS environment).

`Y`:

Check for overlap. If an overlap is found, the `KFPS00348-E` message is output and HiRDB activation is canceled.

`N`:

Do not check for overlap.

`C`:

Check for overlap. The HiRDB reserved port facility is not applied to any overlapping port number.

`W`:

Check for overlap. If an overlap is found, the `KFPS00354-W` message is output. The HiRDB reserved port facility is not applied to any overlapping port number.

Condition

The `pd_registered_port` operand must be specified.

Specification guidelines

- If an overlap in port numbers is detected, HiRDB communication may be affected adversely, resulting for example in the receipt of wrong messages or message send failures.
- If `Y`, `C`, or `W` is specified, process server activation in an NIS or DNS environment may slow down.

Remarks

When HiRDB starts, it checks for overlap between the port numbers included in the `pd_registered_port` operand's range specification and the port numbers specified in the operands listed below. If an overlapping port number is detected, HiRDB displays the `KFPS00348-E` message and cancels the startup processing.

- `pd_name_port` operand

- -p option of the pdunit operand
- pd_service_port operand

Note that overlap checking is not performed on port numbers specified in any other operands.

169) pd_registered_port_level = 0 | 1

Specifies the target range for the HiRDB reserved port facility.

Condition

The pd_registered_port operand must be specified.

Specification guidelines

The following shows the target range depending on the value of this operand (0 or 1):

Port numbers used by HiRDB		Value of pd_registered_port_level	
		0	1
Ports used by HiRDB server	Send port number for server-to-server communication	Y	Y
	Receive port number for server-to-server communication	Y	Y
	Port number specified in pd_name_port, pd_service_port, or pdunit -p operand	N	N
	Send port number for command	N	N
	Receive port number for command	N	N
	Send port number for client	N	Y
	Receive port number for client	Y	Y
Ports used by HiRDB client	Send port number	N	N
	Receive port number	N	N

Legend:

Y: Subject to the HiRDB reserved port facility. A port number specified in the pd_registered_port operand is used.

N: Not subject to the HiRDB reserved port facility.

170) pd_ipc_send_retrycount = *process-to-process-send-retries-count*

~ <unsigned integer>((1-32767))<<200>>(Times)

Specifies the number of times process-to-process communication can be attempted. This operand is related to the `pd_ipc_send_retrysleeptime` operand.

Examples

- `pd_ipc_send_retrycount = 500`
- `pd_ipc_send_retrysleeptime = 2`

When the operands are specified in this way, send is attempted up to 500 times and a 2-second sleep occurs between attempts.

Specification guidelines

- Normally, this operand need not be specified.
- Specifying too large a value for this operand increases the CPU lock rate.

171) pd_ipc_send_retrysleeptime = *process-to-process-send-retry-sleep-time*

~ <unsigned integer>((0-60)) <<0>> (seconds)

Specifies the sleep time between process-to-process send retries.

This operand is related to the `pd_ipc_send_retrycount` operand.

Examples

- `pd_ipc_send_retrycount = 500`
- `pd_ipc_send_retrysleeptime = 2`

When the operands are specified in this way, send is attempted up to 500 times and a 2-second sleep occurs between attempts.

Specification guidelines

- Normally, this operand need not be specified.
- Specifying too large a value for this operand increases communication completion time.

172) pd_ipc_send_count = *server-to-server-send-retries-count*

~ <unsigned integer>((1-32767)) <<60>>

Specifies the number of times a server-to-server send can be performed before the send is completed. A send occurs for up to 5 seconds at each retry. With the default value, send will be performed for up to 60 × 5 seconds = 300 seconds.

Specification guideline

Normally, this operand need not be specified. If transmission timeouts occur frequently, increase this operand's value.

173) pd_ipc_recv_count = *server-to-server-receive-retries-count*

~ <unsigned integer>((1-32767)) <<120>>

Specifies the number of times a server-to-server receive can be performed before receive is completed. Receive occurs for up to 5 seconds at each retry. With the default value, receive is performed for up to 120 × 5 seconds = 10 minutes.

Specification guideline

Normally, this operand need not be specified.

174) pd_ipc_conn_nblock = Y | N

Specifies whether to establish connection in the non-block mode for HiRDB server-to-server communication (communication between units).

`Y`: Establishes the connection in the non-block mode.

`N`: Establishes the connection in the block mode.

Advantage

When you specify `Y` for this operand, you can issue `connect()` system calls in the non-block mode for HiRDB server-to-server communication (communication between units). As a result, the duration for which `connect()` system calls block processes during a LAN error, which used to last dozens of seconds (depending on the OS), can now be shortened to the time specified in the `pd_ipc_conn_nblock_time` operand.

Relationship to other operands

This operand is related to the following operands:

- `pd_ipc_conn_nblock_time`
- `pd_ipc_conn_interval`
- `pd_ipc_conn_count`
- `pd_host_watch_interval`

Specification guidelines

Normally, there is no need to specify this operand.

Remarks

For establishing communication connection between a client and a server, specify the necessary parameter in the client environment definition (`PDNBLOCKWAITTIME` operand).

175) pd_ipc_conn_nblock_time = *connection-establishment-monitoring-time-in-non-block-mode*

~ <unsigned integer>((1-120))<<8>>(seconds)

Specifies whether or not connection is to be established in the non-block mode for HiRDB server-to-server communication (communication between units).

Condition

This operand is valid when `Y` is specified for the `pd_ipc_conn_nblock` operand.

Advantage

If unreasonable communication errors occur frequently, increasing the value specified for this operand may reduce the number of communication errors.

Specification guidelines

Normally, there is no need to specify this operand.

If unreasonable communication errors occur frequently, use the `ping` command of the OS, for example, to measure the communication time between HiRDB servers. If that time is greater than the value specified for this operand, specify a value greater than that communication time for this operand.

Note

Increasing the specification value for this operand increases the timeout duration when communication is disabled. Consequently, other processes may be adversely affected.

Relationship to other operands

This operand is related to the following operands:

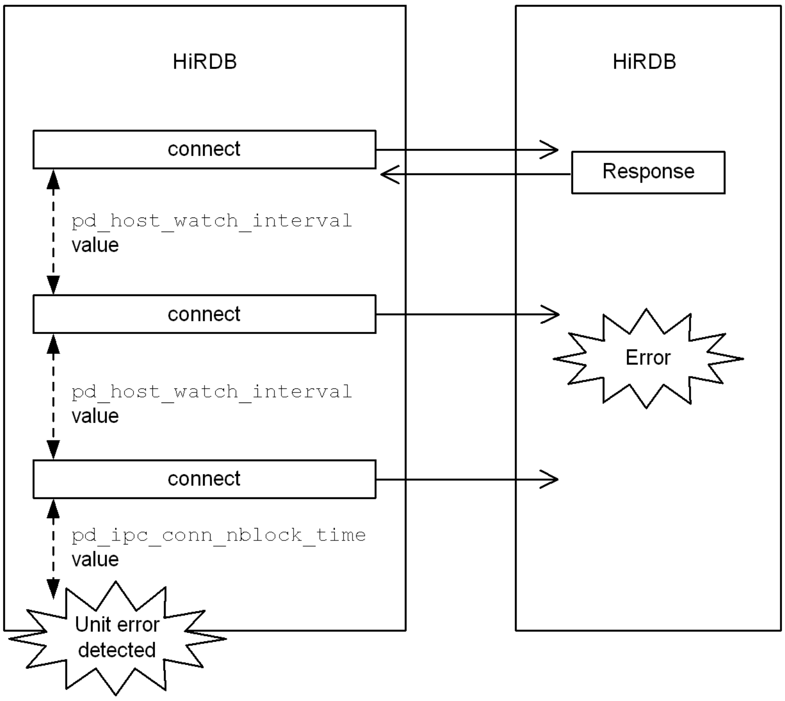
- `pd_ipc_conn_nblock`
- `pd_ipc_conn_interval`
- `pd_ipc_conn_count`

Remarks

HiRDB communicates with other units at a given interval (the value of the `pd_host_watch_interval` operand; the default value is 10 seconds) to monitor the operating status of other units.

If a LAN error occurs immediately after a monitoring-target unit has responded, HiRDB tries to communicate with that unit after a certain wait time (the value of the `pd_host_watch_interval` operand). Thereafter, communication retry is carried out for the duration specified by the `pd_ipc_conn_nblock_time` operand (the default value is 8 seconds). Therefore, the time until the abnormal termination of the monitoring-target unit is detected is the time combining the values of the two operands.

The following figure shows the time line until HiRDB detects the abnormal termination of the monitoring-target unit.


HiRDB
HiRDB
connect
Response
pd_host_watch_interval value
connect
Error
pd_host_watch_interval value
connect
pd_ipc_conn_nblock_time value
Unit error detected

176) pd_ipc_conn_interval = ***connection-establishment-retry-interval***

~ <unsigned integer>((0-50))<<1>>

Specifies the retry interval for establishing connection for sending data. Connection establishment retries are made in the following cases:

- A connection timeout has occurred.
- A shortage has occurred in the `Listen` queue.

Specification guidelines

- Normally, there is no need to specify this operand.
- When specifying this operand, make sure that the following condition is satisfied:

  value-specified-for-this-operand <

value-of-pd_ipc_conn_nblock_time-and-PDNBLOCKWAITTIME-in-client-environment-definition

Relationship to other operands

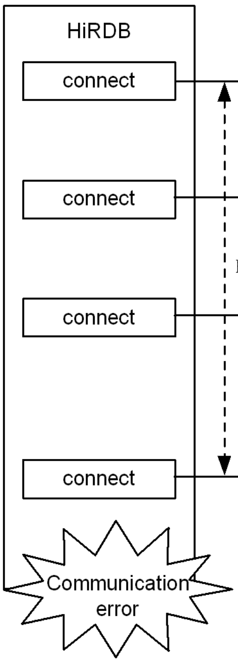
This operand is related to the following operands:

- `pd_ipc_conn_count`
- `pd_ipc_conn_nblock`
- `pd_ipc_conn_nblock_time`

Remarks

The following figure shows examples of retries in non-block mode and block mode communication.

- Retry processing during non-block mode communication (when `pd_ipc_conn_nblock = Y`)

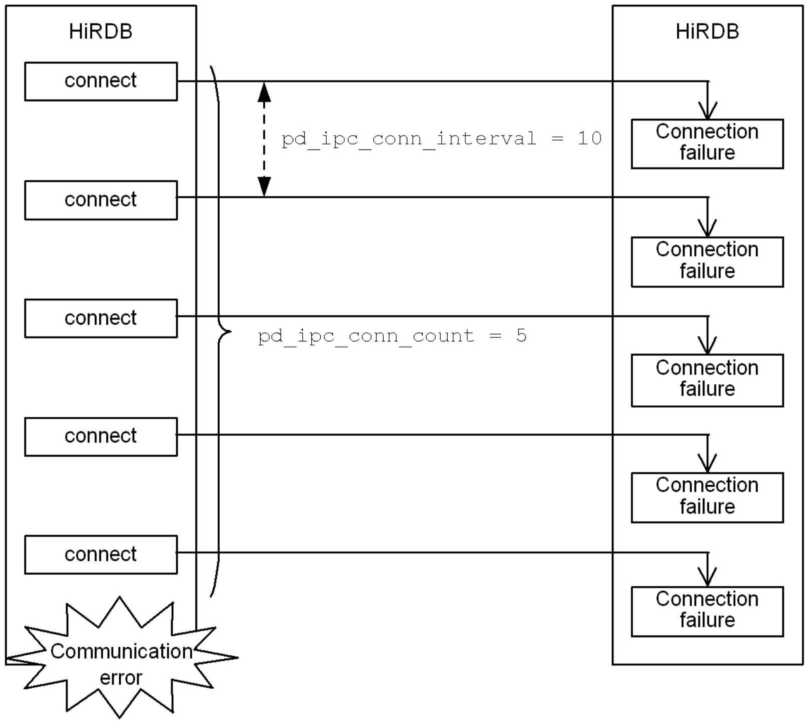


Explanation

With the following specifications, non-block mode connection establishment is retried for 8 seconds at 1-second intervals.

- `pd_ipc_conn_nblock_time = 8`
- `pd_ipc_conn_interval = 1`

- Retry processing during block mode communication (when `pd_ipc_conn_nblock = N`)



Explanation

With the following specifications, non-block mode connection establishment is retried 5 times at 10-second intervals.

- `pd_ipc_conn_interval = 10`
- `pd_ipc_conn_count = 5`

177) pd_ipc_conn_count = ***connection-establishment-retry-count***

~ <unsigned integer>((0-600))<<50>>

Specifies the number of retries for establishing connection for sending data. Connection establishment retries are made in the following cases:

- A connection timeout has occurred.
- A shortage has occurred in the Listen queue.

This operand is related to the `pd_ipc_conn_interval` operand.

Example

- `pd_ipc_conn_interval = 10`
- `pd_ipc_conn_count = 5`

With these specifications, connection establishment is retried every 10 seconds up to 5 times.

Specification guidelines

Normally, there is no need to specify this operand.

Note

For non-block mode connection establishment, the specification of this operand is invalid.

Non-block mode connection establishment is executed when the following conditions are satisfied:

- `Y` is specified for the `pd_ipc_conn_nblock` operand.
- The `PDNBLOCKWAITTIME` operand in the client environment definition is specified.

178) pd_ipc_inet_bufsize = *send-receive-buffer-size-for-server-unit-to-unit-communication*

~ <unsigned integer>((0-262144)) <<16384>> (bytes)

Specifies as a multiple of 4096 a size for the send/receive buffer to be used for unit-to-unit communication within a server (TCP NET domain).

Specification guideline

Normally, this operand need not be specified.

Notes

- If `0` is specified, the value set by the OS is used.
- The maximum TCP buffer size depends on the OS.
- The specified buffer size may not be valid depending on the OS being used. In such a case, the value set by the OS is used.

179) pd_ipc_unix_bufsize = send-receive-buffer-size-for-communication-within-server-unit

~ <unsigned integer>((0-262144))<<16384>>(bytes)

Specifies as a multiple of 4096 a size for the send/receive buffer to be used for communication within a server unit (TCP UNIX domain).

Specification guidelines

Normally, this operand need not be specified.

Notes

- If `0` is specified, the value set by the OS is used.
- The maximum TCP buffer size depends on the OS.
- The specified buffer size may not be valid depending on the OS being used. In such a case, the value set by the OS is used.

180) pd_tcp_inet_bufsize = *send-receive-buffer-size-for-communication-with-HiRDB-client-outside-host-where-HiRDB-server-reside*

~ **<unsigned integer>((0-262144))<<0>>(bytes)**

Specifies as a multiple of 4096 a maximum size for the send/receive buffer to be used for communication with a HiRDB client connected from outside the host where the HiRDB server resides (TCP INET domain).

Specification guidelines

Normally, this operand need not be specified. Specify this operand if one of the following occurs:

- Resending of the data to be sent/received between a HiRDB server and a HiRDB client is observed (specifying this operand can suppress resending)
- The data arrival speed is slower than the data read speed on the receiving side (specifying this operand can prevent the delay caused by waiting for data arrival even when the data volume increases)

Specify for this operand the same value as the buffer size to be used for communication.

Notes

- If `0` is specified, the value set by the OS is used.
- The maximum TCP buffer size depends on the OS.
- The specified buffer size may not be valid depending on the OS being used. In such a case, the value set by the OS is used.

181) pd_tcp_unix_bufsize = *send-receive-buffer-size-for-communication-with-HiRDB-client-inside-host-where-HiRDB-server-resides*

~ **<unsigned integer>((0-262144))<<16384>>(bytes)**

Specifies as a multiple of 4096 a maximum size for the send/receive buffer to be used for communication with a HiRDB client of HiRDB Version 4.0 04-02 or older connected from the host where the HiRDB server resides (TCP UNIX domain).

Specification guidelines

Normally, this operand need not be specified. Specify this operand if one of the following occurs:

- Resending of the data to be sent/received between a HiRDB server and a HiRDB client is observed (specifying this operand can suppress resending)
- The data arrival speed is slower than the data read speed on the receiving side (specifying this operand can prevent the delay caused by waiting for data arrival even when the data volume increases)

Specify for this operand the same value as the buffer size to be used for communication.

Notes

- If 0 is specified, the value set by the OS is used.
- The maximum TCP buffer size depends on the OS.
- The specified buffer size may not be valid depending on the OS being used. In such a case, the value set by the OS is used.

182) pd_utl_buff_size = *utility-communication-buffer-size*

~ <unsigned integer>((8-64))<<32>>(KB)

Specifies in kilobytes the size of each sector of the buffers used for communication by the following utility processes:

- Database load utility
- Database reorganization utility (with -g option specified)
- Database recovery utility
- Database copy utility

Specification guidelines

$$\textit{communication-buffer-size} = \uparrow (\textit{page-size}^{*} \times 2) \div 1024 \uparrow (\text{KB})$$

* Maximum page size of RDAREAs to be processed by the utility.

Operand rules

- Specify a multiple of 4. If the specified value is not a multiple of 4, it is rounded up automatically to a multiple of 4.
- If the specified value is smaller than 32, it will be increased automatically to 32 when the database recovery utility or database copy

utility is executed.

183) pd_sql_send_buff_size = ***buffer-size-for-inter-server-communication-for-SQL-execution***

~ <unsigned integer>((4-32))<<4>>(KB)

This operand is applicable only to a HiRDB/Parallel Server.

Specify the size of each communication buffer (in KB) to be used for sending search result data between HiRDB servers.

Specification guidelines

communication-buffer-size = ↑ *page-size*[*] ÷ 1024 ↑ (Units: KB)

[*] *page-size* = *row-size* × *communication-count-per-case*

- For the data size of variable-length data, use the actual size.
- Because LOB columns and abstract data type columns are excluded, do not include them in the page size.

Note

Increasing the specification value of this operand may increase the amount of memory used.

184) pd_ipc_tcp_nodelayack = Y | N

This operand is applicable only to the AIX 5L version. Do not specify this operand for any other version of HiRDB; if specified in such a case, HiRDB cannot be started.

Specifies whether or not immediate acknowledgment is to be applied. For details about immediate acknowledgment, see the *HiRDB Version 8 Installation and Design Guide*. Immediate acknowledgment is applicable to the AIX 5L V5.2 IY51819 patch-applied version and to AIX 5L V5.3 or later.

`Y`:

Applies immediate acknowledgment. When the transmission target receives packets successfully during data transfer, ACK (acknowledgement) is sent immediately to the transmission source.

`N`:

Does not apply immediate acknowledgment. Even when the transmission target receives packets successfully during data transfer, ACK (acknowledgement) is not sent immediately. Acknowledgement may be sent to the transmission source together with reply data.

Specification guidelines

- If response is poor during communication between HiRDB server and HiRDB client or between HiRDB servers, specifying Y in this operand may improve the situation.
- If the network load is high, specifying N in this operand reduces the number of ACK transmissions, possibly reducing the network load.
- If 1 is specified in the tcp_nodelayack option of the OS parameters, immediate acknowledgment is applied to the entire system regardless of the specification of the pd_ipc_nodelayack operand or of PDNODELAYACK. For details about the tcp_nodelayack option, see the AIX manual.

Relationship to client environment definition

Specify YES in the PDNODELAYACK client environment definition in the following cases; for details about the PDNODELAYACK client environment definition, see the *HiRDB Version 8 UAP Development Guide*:

- During communication between HiRDB server and HiRDB client, the communication socket generated by the HiRDB client is used to receive packets.
- The HiRDB client is on a separate server machine and is the AIX 5L V5.2 IY51819 patch-applied version or AIX 5L V5.3 or later.

2.3.38 Operands related to Java^{TM}

Java operands are specified when a Java stored procedure or Java stored function is used. For details about Java stored procedures and Java stored functions, see the *HiRDB Version 8 UAP Development Guide.*

Note that some platforms on which HiRDB can be run do not support Java stored procedures and Java stored functions. For details, see *Environment supporting Java stored procedures and functions* in the *HiRDB Version 8 System Operation Guide*.

185) pd_java_option = "Java-option" [,*"Java-option"*]...

～ <character string>

Specifies Java virtual option startup options. For details about the startup options, see the documents on Java virtual machines.

Operand rules

- The maximum character string size of a Java option is 255 characters.
- The total character string size of all Java options is 1024 characters.
- Double quotation mark (") cannot be specified within a Java option.
- A maximum of 20 Java options can be specified.

Operand specification example

The following example specifies the initial value of the heap size to be used by a Java virtual machine as 32 megabytes and the maximum value of the heap size as 64 megabytes:

```
pd_java_option = "-Xms32m","-Xmx64m"
```

186) pd_java_routine_stack_size = *stack-size-for-use-by-Java-routine*

~ <unsigned integer>((1024-2146435072))<<131072>>(bytes)

Specifies in bytes the stack size to be used by a Java routine.

Specification guidelines

- Specify a value that is greater than the stack size specified as a Java option.
- If an operation for specifying both the stack size and native method stack area size for a Java virtual machine is available as a startup option of the Java virtual machine, specify for the `pd_java_routine_stack_size` operand a value that is greater than the native method stack size.

Notes

- If a value that is smaller than the stack size specified as a Java option is specified, HiRDB may not run correctly.
- If the stack size being used exceeds the value specified by this operand, HiRDB may not run correctly. However, HiRDB will run correctly if the Java virtual machine detects the stack overflow.

187) pd_java_archive_directory = *"JAR-file-storage-directory"*

~ <pathname> <<$PDDIR/java>>

Specifies the name of the directory for storing JAR files used by Java stored procedures or Java stored functions.

Notes

- The specified directory must be created before installation of JAR files.
- The JAR file storage directory is used only for storing JAR files.
- Store only the installed JAR files in the JAR file storage directory.

Operand rules

- Up to 255 characters can be used for the pathname.

188) pd_java_classpath = *"Java-class-path"*

~ <pathname>

Specifies as an absolute pathname the class path to be used by a Java virtual machine.

The class contained in the path specified by this operand can be referenced from the Java method, which is executed as the processing procedure of a Java stored procedure or Java stored function.

If a class with the same name exists in the path specified by this operand and in the JAR file specified as an external routine of the Java stored procedure or Java stored function, the path specified by this operand takes precedence.

Operand rules

- Up to 1024 characters can be used for the pathname.

189) pd_java_runtimepath = "*Java-Runtime-Environment-root-directory*"

~ <pathname> <<$PDDIR/jre>>

Specifies as an absolute pathname the root directory of the Java Runtime Environment.

Operand rules

- Up to 255 characters can be used for the pathname.

Notes

Because Java Runtime Environment (JRE) is no longer included, you must add this operand or change its value when you upgrade HiRDB from a version earlier than 07-03 to version 07-03 or later. For notes about upgrading, see *Using Java stored procedures and functions* in the *HiRDB Version 8 Installation and Design Guide*.

190) pd_java_libpath = "*Java-virtual-machine-library-directory*"

~ <pathname>

Specifies the directory that stores the library of the Java virtual machine as a relative pathname to the Java Runtime Environment root directory (the value of the `pd_java_runtimepath` operand).

Operand rules

- Up to 255 characters can be used for the pathname.
- The following table shows the default value of this operand

OS	the default value of this operand
HP-UX	`lib/PA_RISC/hotspot`

OS	the default value of this operand
HP-UX (IPF)	`lib/IA64W/server`
Solaris	`lib/sparc/hotspot`
Linux	`lib/i386/server`
Linux (IPF)	`lib/ia64/server`
Linux (EM64T)	`lib/amd64/server`
AIX 5L	`bin/classic`

Operand specification example

This example assumes that the CPU architecture uses a Hot Spot virtual machine compatible with PA-RISC 2.0; it also assumes use of the HP-UX version of Java Runtime Environment 1.2.2:

```
pd_java_libpath = "lib/PA_RISC2.0/hotspot"
```

Note

Do not specify lib/PA_RISC/classic or lib/PA_RISC2.0/classic for the HP-UX version; if you do, HiRDB may not run correctly.

191) pd_java_stdout_file = *"Java-virtual-machine-standard-output-and-standard-error-output-destination-file"*

～ <pathname>

Specifies as an absolute pathname the file to which the standard output and standard error output are to be output in a Java virtual machine. If this operand is omitted, the standard output and standard error output of the Java virtual machine are ignored.

Specification guidelines

Because the size of the file specified by this operand is extremely large, this operand is not normally specified. It is recommended that this operand be specified during debugging of a Java stored procedure or Java stored function. There is no limit to the size of the file that can be specified by this operand.

Note

If there are simultaneous writing attempts from multiple processes, their output contents cannot be guaranteed.

Operand rules

- Up to 255 characters can be used for the pathname.

2.3.39 Operands related to unit structure

192) pdunit -x *host-name*

-u *unit-identifier*

[-d *"HiRDB-directory-name"*]

[-c *host-name*]

[-p *port-number*]

This operand must be specified; it cannot be omitted.

HiRDB/Single Server

The unit structure of a HiRDB/Single Server or a utility special unit is defined. The following items can be specified:

- Host name
- Unit identifier
- HiRDB directory name
- Secondary system host name
- Port number of the utility special unit (or port number of the HiRDB/Single Server)

Specification guidelines

- This operand may be omitted if the system consists of a single HiRDB/Single Server. If this operand is omitted, the HiRDB directory specified by the `pdsetup` command is assumed. However, it cannot be omitted when the system switchover facility is used.
- When a utility special unit is used, the unit information for the HiRDB/Single Server and for the utility special unit is specified in the system common definition of the HiRDB/Single Server. One pdunit operand is specified for the HiRDB/Single Server and another pdunit operand is specified for the utility special unit.
- The unit information for the HiRDB/Single Server and for the utility special unit is specified in the system common definition of the utility special unit. One pdunit operand is specified for the HiRDB/Single Server that uses the utility special unit and another pdunit operand is specified for the utility special unit.
- When the system switchover facility is used, the same definition must be applied to both the primary and secondary systems.

- In the case of mutual system switchover, a multi-HiRDB structure is used (two different HiRDB/Single Servers coexist in a single host), and the same definition cannot be used for two different HiRDB/Single Servers.
- Table 2-2 shows the host name when the pdunit operand's -x option and the pd_hostname operand are specified.

Table 2-2: Host name when pdunit operand's -x option and pd_hostname operand are specified (for HiRDB/Single Server)

Condition	pdunit operand's -x option value	pd_hostname operand value
System switchover facility is not used	Standard host name (host name, IP address, or FQDN)	Default or standard host name (host name, IP address, or FQDN)
	Host name other than the standard host name (host name, IP address, or FQDN)	Standard host name or the host name specified in the pdunit operand's -x option[#1] (host name, IP address, or FQDN)
	Loopback address (localhost or 127.0.0.1)[#2]	Loopback address (localhost or 127.0.0.1)[#1#2]
Monitor mode (IP address inherited)	Relocatable IP address, host name with a relocatable IP address, or FQDN	Standard host name or the host name specified in the pdunit operand's -x option[#1] (host name, IP address, or FQDN) (host name, IP address, or FQDN)
	Loopback address (localhost or 127.0.0.1)[#2]	Loopback address (localhost or 127.0.0.1)[#1#2]
Server mode (IP address inherited)	Relocatable IP address, host name with a relocatable IP address, or FQDN	Primary system's standard host name (host name, IP address, or FQDN)
Monitor mode or server mode (IP address not inherited)[#3]	Primary system's host name (host name, IP address, or FQDN)	Primary system's standard host name (host name, IP address, or FQDN)

Note

If the standard host name is specified as FQDN in the pdunit operand's -x option or in the pd_hostname operand, the standard host name must be defined in the FQDN format.

#1

Use the same format as in the pdunit operand's -x option.

#2

`localhost` can be specified if it can be used as the host name (such as when it is already set in the `hosts` file or name server).

#3

Specify the secondary system's host name (host name, IP address, or FQDN) in the `pdunit` operand's `-c` option.

Note

When a utility special unit is used, HiRDB uses the host name specified by this operand to effect communication between the HiRDB/Single Server and the utility special unit. Therefore, if the host name specification is incorrect, various types of communication errors and timeouts may occur.

-x *host-name*

~ <host name>((1-32 characters))

Specifies in one of the formats listed below the host name of the server machine in which the HiRDB/Single Server is defined. Always use the same specification format for a server machine. For example, once a host name is specified, an IP address must not be specified for a host on the same server machine.

- host name
- IP address
- FQDN
- loopback address (`localhost` or 127.0.0.1)

For specification examples, see *Settings for a DNS server to manage IP addresses* in the *HiRDB Version 8 Installation and Design Guide*.

When specifying the host name

- You cannot specify aliases for host names.
- If you do not use a DNS server, register the host name specified here in the `hosts` file.

When specifying the loopback address

Use the multi-connection address facility or high-speed connection facility. If these functions are not used, the HiRDB server cannot be accessed from other server machines.

-u *unit-identifier*

~ <identifier>((4 characters))

Specifies a unit identifier. The unit identifier specified here is also specified in the pd_unit_id operand in the unit control information definition.

-d "*HiRDB-directory-name*"

~ <pathname of up to 128 characters><</opt/HiRDB_S>>

Specifies the name of the HiRDB directory.

-c *host-name*

~ <host name>((1-32 characters))

Specifies the secondary system's host name in one of the formats listed below. You specify this option for system switchover in which IP addresses are not inherited. Always use the same specification format for a server machine. For example, once a host name is specified, an IP address must not be specified for a host on the same server machine.

- host name
- IP address
- FQDN
- loopback address (localhost or 127.0.0.1)

Do not specify this option for a utility special unit because system switchover cannot be performed for a utility special unit. If this option is specified, a definition error occurs and HiRDB cannot start.

When specifying the host name

- You cannot specify aliases for host names.
- If you do not use a DNS server, register the host name specified here in the hosts file.

When specifying the loopback address

Use the multi-connection address facility or high-speed connection facility. If these functions are not used, the HiRDB server cannot be accessed from other server machines.

-p *port-number*

~ <unsigned integer>((5001-65535))

Specifies the port number of the HiRDB at the other system that uses the local system (the same number as in the pd_name_port operand of the system common definition).

If this option is omitted, the local system's port number (the value specified by the pd_name_port operand of the system common definition) is assumed.

Specification guidelines

- For a HiRDB/Single Server, the port number of the utility special unit that will be used by the HiRDB/Single Server is specified.
- For a utility special unit, the port number of the HiRDB/Single Server that will use the utility special unit is specified. This option enables utility special units with different port numbers to be used.
- This option must be specified when a single utility special unit will be shared in a multi-HiRDB configuration.

Notes

- Specify a number that is not within the range of port numbers assigned automatically by the OS. The range of port numbers assigned automatically by the OS depends on the OS.
- If a port number is specified that is within the range of port numbers assigned automatically by the OS, that number might be in use by another program. If it is, the HiRDB server will not start.

HiRDB/Parallel Server

The unit structure of a HiRDB/Parallel Server is defined. The following items can be specified:

- Host name
- Unit identifier
- HiRDB directory name
- Secondary system host name
- Unit port number

Specification guidelines

- This operand may be omitted if the system consists of a single unit. If this operand is omitted, the HiRDB directory specified by the `pdsetup` command is assumed.
- Information on all units that comprise the HiRDB/Parallel Server must be defined with this operand. For example, if there are three units, three `pdunit` operands must be specified.
- This operand cannot be omitted when the system switchover facility is used.
- When the system switchover facility is used, the same definition must be applied to both the primary and secondary systems.

- In the case of mutual system switchover, a multi-HiRDB structure is used (two different units coexist in a single host), and the same definition cannot be used for two different units.
- Table 2-3 shows the host name when the `pdunit` operand's `-x` option and the `pd_hostname` operand are specified.

Table 2-3: Host name when pdunit operand's -x option and pd_hostname operand are specified (for HiRDB/Parallel Server)

Condition	pdunit operand's -x option value	pd_hostname operand value
System switchover facility is not used	Standard host name (host name, IP address, or FQDN)	Default or standard host name (host name, IP address, or FQDN)
	Host name other than the standard host name (host name, IP address, or FQDN)	Standard host name (host name, IP address, or FQDN)
Monitor mode or server mode (IP address inherited)	Relocatable IP address, host name with a relocatable IP address, or FQDN	Primary system's standard host name (host name, IP address, or FQDN)
Monitor mode or server mode (IP address not inherited)[#1]	Primary system's host name (host name, IP address, or FQDN)	Primary system's standard host name (host name, IP address, or FQDN)

Note

If the standard host name is specified as FQDN in the `pdunit` operand's `-x` option or in the `pd_hostname` operand, the standard host name must be defined in the FQDN format.

#1

- Specify the secondary system's host name (host name, IP address, or FQDN) in the `pdunit` operand's `-c` option.
- The same host name cannot be specified more than once in the `pdunit` operand's `-x` and `-c` options. The following shows examples of correct and incorrect specifications:

Correct examples

```
pdunit -x hostA ... -c hostAA

pdunit -x hostB ... -c hostBB
```

Incorrect examples

```
pdunit -x hostA ... -c hostB

pdunit -x hostB ... -c hostA
```

For an example of HiRDB system definition that uses system switchover in which IP addresses are not inherited, see *C.5 Example 5 (HiRDB/Parallel Server: when the standby system switchover facility is used)*.

-x *host-name*

~ **<host name>((1-32 characters))**

Specifies in one of the formats listed below the host name of the server machine in which the HiRDB/Parallel Server is defined. Always use the same specification format for a server machine. For example, once a host name is specified, an IP address must not be specified for a host on the same server machine.

- host name
- IP address
- FQDN

For specification examples, see *Settings for a DNS server to manage IP addresses* in the *HiRDB Version 8 Installation and Design Guide*.

When specifying the host name

- You cannot specify aliases for host names.
- If you do not use a DNS server, register the host name specified here in the `hosts` file.

-u *unit-identifier*

~ **<identifier>((4 characters))**

Specifies a unit identifier. The unit identifier specified here is also specified in the `pd_unit_id` operand in the unit control information definition of each unit.

-d "*HiRDB-directory-name*"

~ **<pathname of up to 128 characters><</opt/HiRDB_P>>**

Specifies the name of the HiRDB directory.

Notes

- When using the standby-less system switchover (1:1) facility

  Specify the same HiRDB directory name in the normal and back-end server units.

- When using the standby-less system switchover (effects distributed) facility

  Specify the same HiRDB directory name in all units belonging to the same HA group.

-c *host-name*

~ <host name>((1-32 characters))

Specifies the secondary system's host name in one of the formats listed below. You specify this option for system switchover in which IP addresses are not inherited. Always use the same specification format for a server machine. For example, once a host name is specified, an IP address must not be specified for a host on the same server machine.

- host name
- IP address
- FQDN

This option cannot be specified for the units described below. If it is specified, HiRDB cannot start (message `KFPS01896-E` is output).

- Unit to which the standby-less system switchover (1:1) facility is applied. (IP address inheriting does not occur.)
- Unit to which the standby-less system switchover (effects distributed) facility is applied. (IP address inheriting does not occur.)
- Utility special unit (A utility special unit cannot switch systems.)

When specifying the host name

- You cannot specify aliases for host names.
- If you do not use a DNS server, register the host name specified here in the `hosts` file.

-p *port-number*

~ <unsigned integer>((5001-65535))

This option is specified when the system switchover facility is used. Specifies the port number of the unit.

If this option is omitted, the port number of the local system (the value of the `pd_name_port` operand) is assumed.

Specification guidelines

- For the unit in which the system manager is located, the port number specified in `pd_name_port` is specified.
- For a unit in which the system manager is not located, the port number of that unit is specified.
- If there are multiple units in a single server machine, as in a mutual

system switchover configuration, a different port number must be specified for each unit.

Notes

- Specify a number that is not within the range of port numbers assigned automatically by the OS. The range of port numbers assigned automatically by the OS depends on the OS.
- If a port number is specified that is within the range of port numbers assigned automatically by the OS, that number might be being used by another program. If it is, HiRDB cannot start.
- If you use the standby-less system switchover (1:1) facility, specify the same port number for the normal BES unit and the alternate BES unit.
- If you use the standby-less system switchover (effects distributed) facility, specify the same port number for all units belonging to the same HA group.

2.3.40 Operands related to server structure

193) pdstart -t *server-type* [-s server-name]

-x *host-name* | -u *unit-identifier*

[-m *host-name*[,*host-name*]...

[-n *host-name*[,*host-name*]]

[-c *server-name* | -g *HA-group-identifier*]

[-k stls]

This operand must be specified; it cannot be omitted.

Notes

To change a server name that has been specified (with the `-s` option specification), you need to use the database initialization utility to reconstruct the system. Therefore, avoid server names that may have to be changed later.

To change the specification of the `-t`, `-x`, `-u`, `-s`, or `-c` option, you need to perform the following operations. Therefore, avoid specifications that may have to be changed later.

1. Initialization of the system log file (The system log file of the server for which the option was changed must be initialized.)
2. Initialization of the synchronization point dump file (The synchronization point dump file of the server for which the option was changed must be initialized.)

3. Initialization of the status file for unit and the status file for server (The status files for the unit and server for which the options were changed must be initialized.)

HiRDB/Single Server

The server configuration of a HiRDB/Single Server is defined. The following items can be specified:

- Server type
- Server name
- Host name or unit identifier

Specification guideline

- If multiple HiRDB/Single Servers are connected and used, a unique server name must be specified for each HiRDB/Single Server.
- When utility special units are used, their server names must also be different from other server names.
- The `pdstart` operand need not be specified for a utility special unit. However, the `pdstart` operand must be specified for all HiRDB/Single Servers that use the utility special unit.

-t *server-type*

Specifies the server type:

`SDS`: Single server

-s *server-name*

～ <identifier>((1-8 characters))

Specifies the server name of the single server.

-x *host-name*

～ <host name>((1-32 characters))

Specifies the host name specified for the `-x` option of the `pdunit` operand.

-u *unit-identifier*

～ <identifier>((4 characters))

Specifies the identifier of the unit that executes the server. Specifies the unit identifier specified by the `-u` option of the pdunit operand.

When the system switchover facility is used, specification of `-u unit-identifier` instead of `-x host-name` is recommended.

-m *host-name*[,*host-name*]...

～ <host name>((1-32 characters))

Specifies in one of the following formats the host name of the HiRDB/Single Server to which the HiRDB client connects using the *multi-connection address facility*:

- host name
- IP address
- FQDN

For details about the multi-connection address facility, see the *HiRDB Version 8 Installation and Design Guide*.

Specify this option when the network being used between HiRDB clients and HiRDB servers is different from the network being used between the server machines of the HiRDB servers. If a loopback address is specified in the `pd_hostname` operand, this option must be specified in order to access the HiRDB server from another host's HiRDB client.

When specifying the host name

- You cannot specify aliases for host names.
- If you do not use a DNS server, register the host name specified here in the `hosts` file.

Notes

- You can specify a maximum of four this option.
- If you specify multiple host names that are on the same network, the first host name specified is valid.
- If you specify the host name of a server machine not containing a HiRDB/Single Server, that specification is ignored.
- If separate sub-nets are used for the network for connecting HiRDB clients and the network for communication between HiRDB servers, specification of the `-m` option is invalid.
- When you are using a system switchover facility that does not inherit IP addresses, also specify the `-n` option. Specify the host name of the primary system for the `-m` option, and specify the host name of the secondary system for the `-n` option.
- If the system switchover facility that inherits IP addresses is used and a host name other than the standard host or a loopback address is specified in the `pd_hostname` operand, specify the host name with a relocatable IP address in the `-m` option. Also specify the same host name in the `-n` option.

-n *host-name*[*,host-name*]...

~ <host name>((1-32 characters))

Specify this option together with the `-m` option.

When you use the multi-connection address facility, specifies in one of the following formats the host name of the HiRDB/Single Server to which the HiRDB client connects:

- host name
- IP address
- FQDN

Specify this option when the network being used between HiRDB clients and HiRDB servers is different from the network being used between the server machines of the HiRDB servers. When you are using a system switchover facility that does not inherit IP addresses, specify the host name of the secondary system.

When specifying the host name

- You cannot specify aliases for host names.
- If you do not use a DNS server, register the host name specified here in the `hosts` file.

Notes

- You can specify a maximum of four this option.
- If you specify multiple host names that are on the same network, the first host name specified is valid.
- If you specify the host name of a server machine not containing a HiRDB/Single Server, that specification is ignored.
- If separate sub-nets are used for the network for connecting HiRDB clients and the network for communication between HiRDB servers, specification of the `-n` option is invalid.

-c *server-name*

~ <identifier>((1-8 characters))

Omit this option, because it is used only for a HiRDB/Parallel Server.

-g *HA-group-identifier*

~ <identifier>((1-8 characters))

Omit this option because it applies only to a HiRDB/Parallel Server.

-k stls

~ <identifier>((4 characters))

Omit this option because it applies only to a HiRDB/Parallel Server.

HiRDB/Parallel Server

The server configuration of a HiRDB/Parallel Server is defined. The following items can be specified:

- Server type
- Server name
- Host name or unit identifier

Specification guidelines

The following guidelines are for the server configuration:

- Concurrent execution of SQLs can be improved if the individual servers are distributed among separate server machines.
- Concurrent execution of SQLs can be improved if the system manager and front-end servers are defined in server machines connected using the TCP/IP protocol.
- It may in some cases be better to allocate multiple back-end servers to a single server machine, depending on the CPU workloads of the back-end servers.

Note

Only one system manager server and one dictionary server may be specified.

-t *server-type*

Specifies the server type:

`MGR`: System manager

`FES`: Front-end server

`BES`: Back-end server

`DIC`: Dictionary server

-s *server-name*

~ <identifier>((1-8 characters))

Specifies a server name. This option need not be specified if the server type is MGR (system manager).

-x *host-name*

~ <host name>((1-32 characters))

Specifies the host name specified for the `-x` option of the `pdunit` operand.

-u *unit-identifier*

~ <identifier>((4 characters))

Specifies the identifier of the unit that executes the server. Specifies the unit identifier specified by the `-u` option of the pdunit operand.

When the system switchover facility is used, specification of `-u unit-identifier` instead of `-x host-name` is recommended.

-m *host-name*[,*host-name*]...

~ <host name>((1-32 characters))

Specifies in one of the following formats the host name of the front-end server to which the HiRDB client connects using the :

- host name
- IP address
- FQDN

For details about the multi-connection address facility, see the *HiRDB Version 8 Installation and Design Guide*.

Specify this option when the network being used between HiRDB clients and HiRDB servers is different from the network being used between the server machines of the HiRDB servers.

When specifying the host name

- You cannot specify aliases for host names.
- If you do not use a DNS server, register the host name specified here in the `hosts` file.

Notes

- You can specify a maximum of four this option.
- If you specify multiple host names that are on the same network, the first host name specified is valid.
- If you specify the host name of a server machine not containing a front-end server, that specification is ignored.
- If separate sub-nets are used for the network for connecting HiRDB clients and the network for communication between HiRDB servers, specification of the `-m` option is invalid.

- When you are using a system switchover facility that does not inherit IP addresses, also specify the `-n` option. Specify the host name of the primary system for the `-m` option, and specify the host name of the secondary system for the `-n` option.
- If the system switchover facility that inherits IP addresses is used and a host name other than the standard host is specified in the `pd_hostname` operand, specify the host name with a relocatable IP address in the `-m` option. Also specify the same host name in the `-n` option.

-n *host-name*[,*host-name*]...

~ <host name>((1-32 characters))

Specify this option together with the `-m` option.

If you use the multi-connection address facility, specifies in one of the following formats the host name of the front-end server to which the HiRDB client connects:

- host name
- IP address
- FQDN

Specify this option when the network being used between HiRDB clients and HiRDB servers is different from the network being used between the server machines of the HiRDB servers. When you are using a system switchover facility that does not inherit IP addresses, specify the host name of the secondary system.

When specifying the host name

- You cannot specify aliases for host names.
- If you do not use a DNS server, register the host name specified here in the `hosts` file.

Notes

- You can specify a maximum of four this option.
- If you specify multiple host names that are on the same network, the first host name specified is valid.
- If you specify the host name of a server machine not containing a front-end server, that specification is ignored.
- If the network for connecting HiRDB clients is separated from the communication network between HiRDB servers by a subnet, specification of the `-n` option is invalid.

-c *server-name*

~ <identifier>((1-8 characters))

This option is related to the standby-less system switchover (1:1) facility. Specify for this option the name of the alternate BES name of the back-end server specified by the `-s` option.

If you specify the `-c` option, all of the following conditions must be satisfied. Otherwise, HiRDB cannot be started. Here, it is assumed that the `-u` option is specified for the `pdstart` operand.

1. `BES` must be specified for the `-t` option.
2. The normal BES unit and the alternate BES unit must not contain any servers except back-end servers.
3. Specify the alternate BESs for all of the back-end servers in the normal BES unit (the unit specified in the `-u` option). Additionally, you must specify those BESs in the same unit (alternate BES unit).
4. You cannot specify duplicate server names in the `-c` option in the normal BES unit.
5. The number of back-end servers in the normal BES unit must be the same as the number of back-end servers in the alternate BES unit.
6. If you specify the `-c` option for the `pdstart` operand of the alternate BES, you must specify for this option the back-end servers inside the normal BES unit.

-g *HA-group-identifier*

~ <identifier>((1-8 characters))

When you use the standby-less system switchover (effects distributed) facility, this option specifies an *HA group* identifier. The HA group identifier is a set of units that becomes the destination to which the server specified by the `-s` option is to be moved, and must be specified in the `pdhagroup` operand. For details about HA groups, see the *HiRDB Version 8 System Operation Guide*.

When you specify this option, all of the following conditions must be satisfied. Otherwise, HiRDB cannot start.

- Condition for the server to be specified in the -s option
  - `BES` is specified for the `-t` option. (A back-end server must be specified.)
  - The -c option is not specified. (An alternate back-end server cannot be specified for this server.)

- Condition for the unit to be specified in the -u option
  - The normal unit belongs to the HA group specified by the `-g` option.
  - The normal unit is comprised of back-end servers only.
  - An HA group is specified as the moving destination for the servers comprising the normal unit.
- Condition for the HA group to be specified in the -g option
  - All units within the HA group belong to the same network segment.
  - The standby-less system switchover (effects distributed) facility is being applied. (`pd_ha_agent = activeunits` is specified.)
  - At least one host back-end server exists inside the unit.
  - The unit is comprised of back-end servers only.
  - All servers within the unit belong to an HA group.

-k stls

～ <identifier>((4 characters))

Specify this option when using a *recovery-unnecessary front-end server*. To use a recovery-unnecessary front-end server, you need the HiRDB Non Recover FES.

If any of the following conditions is satisfied, HiRDB cannot start.

- Specification for this option contains an error.
- The server specified for the `-t` option is not a front-end server.
- This option is specified for a unit containing a server that is not a front-end server.
- HiRDB Non Recover FES is not set up.

Notes

When Real Time SAN Replication based on the log-only synchronous method is used, the recovery-unnecessary front-end server cannot be used at the log application site. When the `-k stls` option is specified, the `KFPS04689-W` message is displayed.

Relationship to other operands

- If you specify this option, re-estimate the value of the `pd_log_max_data_size` operand.
- If this option is specified for a front-end server, that front-end server's unit in the `pd_start_skip_unit` operand is ignored, if specified.

- The system switchover facility is not applicable to a recovery-unnecessary front-end server unit. If your system uses the system switchover facility, make sure that `nouse` is specified in the `pd_ha_unit` operand for the recovery-unnecessary front-end server unit.

2.3.41 Operands related to the global buffers

194) pdbuffer *-a buffer-name*

{-r *RDAREA-name*[,*RDAREA-name*]...|

-b *RDAREA-name*[,*RDAREA-name*]... |

-o |

-i *authorization-identifier.index-identifier* }

-n *buffer-sectors-count* [-l *buffer-size*]

[-m *maximum-concurrently-executable-prefetches-count*]

[-p *maximum-batch-input-pages-count*]

[-w *updated-pages-output-rate-during-deferred-write-trigger*]

[-c]

[-y *update-buffer-sectors-count-for-deferred-write-trigger-event*]

Specifies the RDAREAs to which a global buffer is to be allocated. A global buffer is an area for storing data during input of table data to or output of table data from RDAREAs; such a buffer is allocated in the shared memory.

Global buffers must be allocated to all RDAREAs. An SQL statement or the `pdopen` command cannot be executed for an RDAREA to which a global buffer has not been allocated. For details about how to design global buffers, see the *HiRDB Version 8 Installation and Design Guide*.

Operand rules

You can specify the `pdbuffer` operand up to 2,000,000 times. For a HiRDB/ Parallel Server, you can specify this operand up to 2,000,000 times per server. However, the overall limit is 2,147,483,647.

Tuning the specified values

For details about how to tune global buffers, see the *HiRDB Version 8 System Operation Guide.*

Operand default value

If the `pdbuffer` operand is omitted, global buffers are allocated under the following conditions:

- Global buffer name: `gbuf00`
- Global buffer type: As specified by the `-o` option
- Global buffer size: 6 MB
- Number of buffer sectors: 6 MB ÷ *maximum-RDAREA-page-size*

-a *buffer-name* ~ <identifier>((1-16 characters))

Specifies a name for the global buffer. The same name cannot be repeated.

-r *RDAREA-name*[,*RDAREA-name*]...

~ <identifier>((1-30 characters))

Specifies the names of RDAREAs to which the global buffer is to be allocated. The names of the following types of RDAREAs can be specified:

- Master directory RDAREA
- Data dictionary RDAREAs
- Data directory RDAREA
- User RDAREAs
- Data dictionary LOB RDAREAs*
- User LOB RDAREAs*
- Registry RDAREA
- Registry LOB RDAREA*
- List RDAREA

* It is recommended that a LOB global buffer defined with the `-b` option be allocated to a LOB RDAREA (by specifying both the `-r` and `-b` options).Allocating LOB global buffers improves performance.

Specification guidelines

When RDAREAs with the same or similar page sizes are allocated to the same global buffer, the number of inputs/outputs can be reduced. However, in the following cases, the number of inputs/outputs can be reduced if RDAREAs with the same page size are allocated to different global buffers:

- RDAREAs that have tables with different purposes
- RDAREAs that involve many random accesses and RDAREAs that involve sequential accesses

Operand rules

- If the name of an RDAREA includes characters other than single-byte alphanumeric characters, the name must be enclosed in quotation marks (").
- When an RDAREA name is enclosed in quotation marks ("), the name becomes case sensitive (lowercase characters are distinguished from uppercase characters). However, if an RDAREA name is not enclosed in quotation marks, all the characters are handled as uppercase characters.
- A maximum of 3200 RDAREAs may be defined for a single global buffer.

-b *RDAREA-name*[,*RDAREA-name*]...

~ <identifier>((1-30 characters))

Specifies the names of RDAREAs to which a LOB global buffer is to be allocated. The names of the following types of RDAREAs can be specified:

- Data dictionary LOB RDAREAs
- User LOB RDAREAs
- Registry LOB RDAREA

A LOB RDAREA specified here must also be specified in the `-r` option. If the `-b` option only is specified, the LOB RDAREA cannot be accessed. A specification example follows:

Examples

Global buffers are allocated to a LOB RDAREA (RDLOB01) by specifying the `-r` and `-b` options:

```
pdbuffer -a gbuf01 -r RDLOB01 -n 1000
pdbuffer -a gbuf02 -b RDLOB01 -n 1000
```

Remarks

- The LOB RDAREA consists of a directory portion and a data portion. These portions are managed by different global buffers. The directory portion is cached in the global buffer with the `-r` option and the data portion is cached in the global buffer with the `-b` option. Therefore, both the `-r` (or `-o`) and `-b` options must be simultaneously specified.
- Because the global buffer with the `-r` option and the global buffer with the `-b` option are used for different purposes, their sizes must be separately estimated. Because the global buffer with the `-r` option caches only the directory portion, it can be relatively small. In contrast,

the global buffer with the `-b` option caches the data portion of the LOB RDAREA. Therefore, estimate the size of this global buffer by taking into consideration both the available memory size and hit rates. For the method of estimating global buffer sizes, see the *HiRDB Version 8 Installation and Design Guide*.

Specification guidelines

The following types of LOB RDAREAs can be specified:

- LOB RDAREAs for storing plug-in indexes
- LOB RDAREAs for storing a small volume of data that is accessed frequently

It is recommended that a single LOB RDAREA be allocated to a single LOB global buffer.

Operand rules

- If the name of an RDAREA includes characters other than single-byte alphanumeric characters, the name must be enclosed in quotation marks (").
- When an RDAREA name is enclosed in quotation marks ("), the name becomes case sensitive (lowercase characters are distinguished from uppercase characters).However, if an RDAREA name is not enclosed in quotation marks, all the characters are handled as uppercase characters.
- A maximum of 3200 RDAREAs may be defined for a single global buffer.

Notes

LOB global buffers do not use the prefetch facility or the deferred write trigger facility. Therefore, the `-m, -p,` and `-w` options need not be specified.

-o

Specifies that this global buffer is to be allocated to all RDAREAs that are not specified in the `-r` option. The `-o` option may be specified only once. If it is specified more than once, the first specification is used.

-i *authorization-identifier.index-identifier*

authorization-identifier ~ **<identifier>((1-8 characters))**

index-identifier ~ **<identifier>((1-30 characters))**

Specifies the name (*authorization-identifier.index-identifier*) of an index to which the global buffer is to be allocated as an index global buffer.

Specification guidelines

Specify an index that is used frequently. When a global buffer is allocated to a frequently used index, the memory residency of the index page increases, and as a result, the number of inputs and outputs can be reduced.

This effect can be especially large if an index defined as a cluster key or unique key is allocated to the global buffer. Note that because the index identifier of a cluster key is determined by HiRDB, check the index identifier by searching the dictionary table (`INDEX_NAME` column of the `SQL_INDEXES` table) after a table has been defined.

If all indexes are allocated to the global buffer for indexes, the overall efficiency of the global buffer declines. Therefore, carefully select the indexes to be allocated to the global buffer according to the memory size.

Operand rules

- Each *authorization-identifier.index-identifier* to be specified must be unique.
- When an authorization identifier or index identifier includes characters other than single-byte alphanumeric characters, the authorization identifier or index identifier must be enclosed in quotation marks (").
- When an authorization identifier or index identifier is enclosed in quotation marks ("), the name becomes case sensitive (lowercase characters are distinguished from uppercase characters).However, if an authorization identifier or index identifier is not enclosed in quotation marks, all the characters are handled as uppercase characters.

-n *buffer-sectors-count*

~ **<unsigned integer>**

- 32-bit mode: **((4-460000))**
- 64-bit mode: **((4-1073741824))**

Specifies a sectors count for the global buffer. This option is required.

Specification guidelines

- If an unnecessarily large value is specified, the number of inputs/outputs will decrease but the overhead for buffer retrieval will increase.
- Because global buffers are allocated in the shared memory, allocating an unnecessarily large area will cause frequent paging during use of other memory, resulting in degraded performance.
- Output operations to the database may become concentrated depending on the synchronization point acquisition timing and the updated pages rate in the global buffer. Therefore, the balance of I/O operations must be taken into consideration as well.

- Note that if the number of buffer sectors is too large, it may not be possible to allocate shared memory.
- The following table can be used to determine an appropriate global buffer sectors count:

Condition		Appropriate global buffer sectors count
Global buffer to which specification of `-r` or `-o` option is applicable[1]	HiRDB/Single Server	Number of concurrently occurring SQL processing requests × number of pages used by one SQL (3-6)
	HiRDB/Parallel Server	Number of concurrently executing users × average number of concurrently accessed tables per transaction × 3 × n[2]
Global buffer to which specification of `-b` option is applicable		Number of data pages stored in LOB RDAREA × residency degree[3] (%)
Global buffer to which specification of `-i` option is applicable[1]		Number of pages storing index × residency degree[4] (%)

[1] The minimum number of buffer sectors necessary is explained below. Specify a value that at least equals this value for the buffer sector count. If a value smaller than this value is specified, a buffer shortage may cause an SQL error.

- HiRDB/Single Server

  number-of-SQL-process-requests-that-occur-concurrently × 4
- HiRDB/Parallel Server

  number-of-concurrently-executing-users × *average-number-of-concurrently-accessed-tables-inside-each-transaction* × 4

[2] A buffer sectors count derived with *n*=1 cannot increase the buffer hit rate. A count needs to be specified that takes into consideration how many extra sectors the coefficient should provide (how high the buffer hit rate should be).

[3] Although a residency degree of 1 (100%) is desirable, the value that is used must take into account the memory capacity, data access frequency, etc. For details about the total number of pages in a user LOB RDAREA and the total number of pages in a registry RDAREA, see the *HiRDB Version 8 Installation and Design Guide.*

[4] Although a residency degree of 1 (100%) is desirable, the value that is used must take into account the memory capacity and importance of the index. For details about the number of pages for storing an index, see the *HiRDB Version 8*

Installation and Design Guide.

Tuning the specified value

The value should be set so that the buffer hit rate is at least 80% for HiRDB jobs. The buffer hit rate can be determined as follows:

- With the statistics analysis utility (check the *updated buffer hit rate and referenced buffer hit rate* in the statistical information on the global buffer)
- By checking the header `HIT` from the `pdbufls` command.

-l *buffer-size*

~ <unsigned integer>((even number between 4 and 30))

Specifies in kilobytes the size of the global buffer. Specify this option with the `-n` option.

Specification guidelines

Normally, omit this option. If this option is omitted, the maximum page size of the RDAREA allocated to this global buffer is used as the buffer size. For a HiRDB/Parallel Server, the maximum page size of the RDAREA inside each server is used as the buffer size, and consequently, the buffer size may be different for each server.

In the following cases, consider changing the specification value:

- A value on the larger side should be set when an RDAREA with a page size exceeding the value that would be set in this option will be added later or the maximum RDAREA page size will be increased through reinitialization. However, if HiRDB can be stopped, there is no need to change the specification value because the maximum page size will be set for the buffer size during the subsequent HiRDB startup.

Operand rules

- If the value specified for this option is smaller than the maximum page size of the RDAREAs, the latter value will be used as the buffer size.
- If an odd number is specified for this option, the actual buffer size will be 1 greater than the specified value.

Note

When the value of this option is changed, the new value will not go into effect until HiRDB is started normally. When HiRDB is restarted, the buffer size that was in effect during the last operation (the size before the change) is used.

-m *maximum-concurrently-executable-prefetches-count*

~ <unsigned integer>((0-95000))<<0>>

Specifies the maximum number of prefetch facilities that can be used concurrently. The prefetch facility can reduce the input/output time when a large volume of data is retrieved using a character special file.

When `0` is specified or this option is omitted, the prefetch facility will not work. To use the prefetch facility, a value of at least 1 must be specified.

Specification guidelines

The number of times the SQLs to which the prefetch facility will be applied will be executed concurrently in the RDAREA table allocated to this global buffer should be specified. The SQLs to which the prefetch facility will be applied are listed below (note, however, that an executions count of 2 will be used for No. 3):

1. `SELECT, UPDATE,` or `DELETE` statement (excluding = condition and `IN` condition) that does not use an index.
2. `SELECT, UPDATE,` or `DELETE` statement (excluding = condition and `IN` condition) that performs ascending-order retrieval* using an index
3. `SELECT, UPDATE,` or `DELETE` statement (excluding = condition and `IN` condition) that performs ascending-order retrieval* using a cluster key

* In the order specified in the index definition in the case of a multicolumn index

Notes

When the prefetch facility is used, a buffer dedicated for batch input is allocated in the shared memory for global buffers, separately from the global buffers. Consequently, the size of the shared memory must be estimated again.

For details about the prefetch facility and the formula for determining the size of the shared memory to be used by global buffers, see the *HiRDB Version 8 Installation and Design Guide.*

-p *maximum-batch-input-pages-count*

~ <unsigned integer>((2-256))<<32>>

Specifies the maximum number of pages that can be input in a batch by the prefetch facility. This option is valid only when a value of at least 1 is specified for the `-m` option.

Specification guidelines

The value to be specified is based on the size of the shared memory and its reduction cost performance so that it satisfies the following formula:

$a \times b$ = 64-128 (KB)

a: Page size of the RDAREA that stores the data or index to be prefetched.

b: Maximum number of pages to be input in a batch.

-w *updated-pages-output-rate-during-deferred-write-trigger*

~ <unsigned integer>((0-100))<<20>>

Specifies as a percentage the updated pages output rate at a deferred write trigger. For details about the deferred write trigger, see the *HiRDB Version 8 Installation and Design Guide.*

If `0` is specified in this option, updated pages are not output at a deferred write trigger.

Specification guidelines

- The value to be specified is determined by using the statistics analysis utility to check the number of inputs/outputs and the updated pages hit rate for each global buffer. A low output rate should be specified for a global buffer with a high updated pages hit rate, and a high output rate should be specified for a global buffer with a low updated pages hit rate.
- If an unnecessarily large value is specified, frequent updates will result in a large number of input/output operations. Moreover, the number of pages that must be written into the database during delayed write will increase, reducing the throughput. Conversely, if an unnecessarily small value is specified, the number of pages that must be written into the database during a synchronization point dump may increase. A value should be selected that minimizes the number of times the same page is written into the database during synchronization point dump output intervals.
- If all pages are in the global buffer and multiple transactions update the same page frequently, specify `0` for this operand. Specifying `0` can reduce the total number of pages that are output to the database within the synchronization point interval.

-c

This option is related to the standby-less system switchover facility. Therefore, this option is ignored, even if you specify it when you are not using this facility.

- Standby-less system switchover (1:1) facility

  The specification for this option determines how alternating global buffers are allocated. For the method of allocating the alternating global buffers, see the *HiRDB Version 8 System Operation Guide*.

Note that if you omit both the `-c` and `-o` options, an alternate BES unit cannot be started.

- Standby-less system switchover (effects distributed) facility

  The specification of this operand determines the method of allocating global buffers to the RDAREAs or indexes of the back-end server that is the target of the standby-less system switchover (effects distributed) facility. For details about how to allocate global buffers when the standby-less system switchover (effects distributed) facility is being used, see the *HiRDB Version 8 System Operation Guide*.

  For a unit that is the target of the standby-less system switchover (effects distributed) facility, a global buffer with the `-o` option specified, but not the `-c` option, is ignored.

-y *update-buffer-sectors-count-for-deferred-write-trigger-event*

~ <unsigned integer>((2-2147483647))

Specifies the deferred write trigger in terms of a number of update buffer sectors. When the number of update buffer sectors reaches the specified value, the updated pages are written to the disk. Use this option to set the deferred write trigger for each global buffer.

Specification guidelines

Normally omit this operand. Deferred write processing may not always be completed within the synchronization point dump acquisition interval. You can specify this operand for such cases if you want to shorten the writing time by reducing the number of updated buffers and reducing slightly the updated buffer hit rate. A guideline for this operand's value is to use 50% (the initial value set by HiRDB) or determine the operand's value by referring to *Tuning the synchronization point processing time when deferred write processing is used* in the *HiRDB Version 8 System Operation Guide*.

Operand rules

If the value specified in this option is greater than the number of global buffer sectors, the number of the global buffer sectors is used.

Notes

If the value of this option is too small, the number of times deferred write processing is executed increases, resulting in an increase in the workload; therefore, it is recommended that you specify an appropriate value in accordance with the specification guidelines.

Relationships to other operands

This operand has the following relationships with the pd_dbbuff_rate_updpage operand:

- The value set for the pd_dbbuff_rate_updpage operand applies to all the global buffers.
- The value of the pdbuffer operand's -y option applies to each global buffer.
- The pdbuffer operand's -y option takes precedence.
- If the pdbuffer operand's -y option is omitted, the number of update buffer sectors for deferred write trigger event depends on the specification of the pd_dbbuff_rate_updpage operand, as shown below:

pd_dbbuff_rate_updpage operand specification	Number of update buffer sectors for deferred write trigger event
Specified	Number of global buffer sectors x pd_dbbuff_rate_updpage operand value
Omitted	Determined automatically by HiRDB

2.3.42 Operands related to HA groups

195) pdhagroup -g *HA-group-identifier* -u *unit-identifier*[,*unit-identifier*]...

When you are using the standby-less system switchover (effects distributed) facility, use this operand to define an *HA group*. For details about HA groups, see the *HiRDB Version 8 System Operation Guide*.

Note that this operand cannot be specified more than once.

-g *HA-group-identifier*

~ <identifier>((1-8 characters))

Specifies an HA group identifier. Specify an identifier that is unique to an HA group within the system.

-u *unit-identifier*[,*unit-identifier*]...

~ <identifier>((4 characters))

Specifies the unit identifier for the unit that comprises an HA group. The number of units that can be specified is between 2 and 32. Specifying duplicate units causes an error. The units to be specified in this option must satisfy all of the following conditions:

- All units within the HA group belong to the same network segment.

- The standby-less system switchover (effects distributed) facility is being applied. (`pd_ha_agent = activeunits` is specified.)
- At least one host back-end server exists inside the unit. (An accepting-only unit cannot be defined.)
- The number of host back-end servers within the unit plus the maximum number of guest back-end servers that can be accepted (value specified for the `pd_max_act_guest_servers` operand) does not exceed 34.
- The unit is comprised of back-end servers only.
- All servers within the unit belong to an HA group.

2.3.43 Operands related to statistical information

196) pdstbegin

- Operand specification format in a HiRDB/Single Server

`pdstbegin [-k` *statistical-information-type*`[,`*statistical-information-type*`]...]`

`[-m` *interval*`]`

- Operand specification format in a HiRDB/Parallel Server

`pdstbegin [-k` *statistical-information-type*`[,`*statistical-information-type*`]...]`

`[-m` *interval*`]`

`[{-x` *host-name* `| -u` *unit-identifier*`}]`

`[{-a | -s` *server-name*`[,`*server-name*`]...}]`

This operand is specified in order to begin collecting statistical information at the time HiRDB is started. The `pdstend` command is entered in order to stop collection of statistical information. The statistical information is output to the statistics log file.

-k *statistical-information-type*[,*statistical-information-type*]...

~ **<<sys>>**

Specifies the type of statistical information that is to be output. The `pdls -d sty` command can be used to check the types of statistical information specified for output.

Statistical information type (-k option specification)	Type of statistical information output	Statistical information output trigger
sys	Statistical information on system operation	Statistical information is output at the interval specified in the -m option.
uap	Statistical information on UAPs	Statistical information is output during connection to and disconnection from HiRDB.
sql	Statistical information on SQLs	Statistical information is output when SQL execution starts and ends.
buf	Statistical information on global buffers[1]	Statistical information is output at synchronization points.
fil	Statistical information on HiRDB files related to database manipulations	
dfw	Statistical information on deferred write processing	
idx	Statistical information on indexes	
sop	Statistical information on SQL static optimization	Statistical information is output when no hit occurs on an SQL object buffer during preprocessing of a dynamic SQL or static SQL.
dop	Statistical information on SQL dynamic optimization	Statistical information is output when an SQL statement other than FETCH or CLOSE is executed.
pcd	Statistical information related to SQL object execution	Statistical information is output when an SQL object is executed.
sqh	Statistical information on SQL statement statistics[2]	Statistical information is output during PREPARE and when an embedded SQL is executed.
obj	Statistical information on SQL object transmission[3]	Statistical information is output when an SQL object is executed.
fsv	Statistical information on foreign server option[3]	Statistical information is output when a transaction is terminated.
hba	Statistical information on foreign server usage[3]	Statistical information is output when an SQL statement is executed for a foreign server.

[1] This statistical information is collected at synchronization points, and the information between synchronization points is edited. Consequently, statistical information cannot be collected unless at least two synchronization points occur.

To reliably collect this statistical information, a synchronization point must be triggered by executing the pdlogswap or pdlogsync command immediately before executing the pdstend command.

[2] The statistical information on SQL statement statistics is output when the statistical information on SQLs (sql specification) is being output.

[3] This statistical information is only for a HiRDB/Parallel Server, and is not output for a HiRDB/Single Server.

-m interval

~ **<unsigned integer> ((1-1440)) <<10>>**

This option is specified when statistical information on system operation (sys specified) is to be collected.

Specifies in minutes the interval at which the statistical information on system operation is to be output to the statistics log file.

-x *host-name*

~ **<host name> ((1-32 characters))**

For a HiRDB/Parallel Server, specifies a host name specified in the -x option of the pdunit operand in order to select the units for which statistical information is to be output. Specify the host name of the unit for which statistical information is to be output.

If the standby-less system switchover (effects distributed) facility is being applied to this unit, the statistical information on the guest back-end server being accepted is also output.

-u *unit-identifier*

~ **<identifier> ((4 characters))**

In the case of a HiRDB/Parallel Server, specifies the unit name of a unit for which statistical information is to be output. This option is specified when it is necessary to limit the units for which statistical information is to be output.

If the standby-less system switchover (effects distributed) facility is being applied to the unit on this host, the statistical information on the guest back-end server being accepted is also output.

-a

In the case of a HiRDB/Parallel Server, specifies that statistical information is to be output for all servers. This option is specified when it is not necessary to limit the units for which statistical information is to be output.

-s *server-name*[,*server-name*]...

~ <identifier> ((1-8 characters))

In the case of a HiRDB/Parallel Server, specifies the server names of the servers for which statistical information is to be output. This option is specified when it is necessary to limit the servers for which statistical information is to be output. Some types of statistical information are not output for all server types, as indicated in the following table:

Statistical information type (-k option specification)	Server type		
	FES	DS	BES
`sys`	Y	Y	Y
`uap`	Y	—	—
`sql`	Y	—	—
`buf`	—	Y	Y
`fil`	—	Y	Y
`dfw`	—	Y	Y
`idx`	—	Y	Y
`sop`	Y	—	—
`dop`	Y	—	—
`pcd`	Y	Y	Y
`sqh`	Y	—	—
`obj`	—	Y	Y
`fsv`	—	—	Y
`hba`	—	—	Y

Y: Statistical information is output.

—: Statistical information is not output.

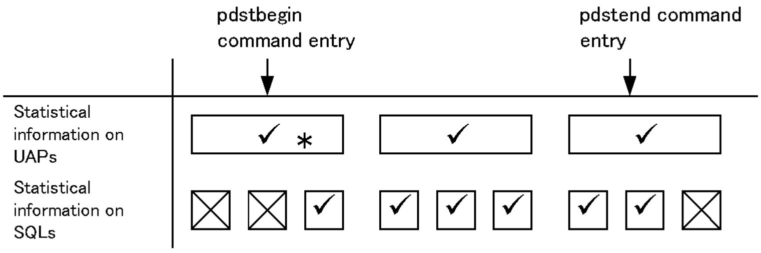
Relationship to other operands

When this operand is specified, specification of the `pd_statistics` operand is ignored.

Notes

- The `pdstbegin` operand can be specified only once. If it is specified more than once, the first time it is specified is valid and the subsequent specifications are ignored.

- When the pdstbegin operand is specified, the specified statistical information is output until HiRDB is terminated or until the pdstend command is entered.
- If you use the standby-less system switchover (1:1) facility, the specification for this operand is invalid when the system is being switched to an alternate BES unit.
- Note the following if you use the standby-less system switchover (effects distributed) facility: During a normal HiRDB startup, each server follows the value specified for this operand. During a HiRDB restart (including a system switchover), the value specified for this operand is invalid. During a restart, the previous statistical information collection state is inherited. During a system switchover, the statistical information prior to the switchover is inherited.
- If no server has started in the unit, unit statistical information is not acquired.
- When Real Time SAN Replication based on the log-only synchronous method is used, statistical information cannot be acquired at the log application site. If this operand is specified, the KFPS04689-W message is displayed.
- Depending on when the pdstbegin and pdstend commands are entered, the statistical information on UAPs may not match the statistical information on SQLs. The relationship between when the pdstbegin and pdstend commands are entered and the statistical information that is output is shown as follows:



✓ : Statistical information is output.
x : Statistical information is not output.
Note: Statistical information is not output under an OLTP environment.

- If the HiRDB system, unit, or server is terminated (including abnormal termination) and is then started again, the statistical information collection state may not be inherited. The following table shows whether the statistical information collection state is inherited when the HiRDB system, unit, or server is started.

Start mode	Statistical information collection environment	Start condition			
		HiRDB start	Unit start	Server start	
				Standby-less system switchover (effects distributed) facility is not used	Standby-less system switchover (effects distributed) facility is used
Normal start	Statistical information is being collected with the `pdstbegin` operand specified.	Y	Y	N*	Y
	Statistical information is being collected by executing the `pdstbegin` command.	N*	N*	N*	N*
Restart	Statistical information is being collected with the `pdstbegin` operand specified.	Y	Y	—	I
	Statistical information is being collected by executing the `pdstbegin` command.	N*	N*	—	Y

Y: Statistical information collection state is inherited.

I: Statistical information collection state is inherited. When the `pdstbegin` command is executed, the collection state of the statistical information specified by this command is inherited.

N: Statistical information collection state is not inherited.

—: Not applicable.

* To collect statistical information, you must execute the `pdstbegin` command after starting the HiRDB system.

197) pdhibegin -k *statistics-type*[,*statistics-type*]...

Specifies the types of statistical information to be collected beginning at the time of HiRDB startup. When this operand is specified, statistical information is output continuously to the system log file until HiRDB is terminated.

-k *statistics-type*

Specifies the type of statistical information to be output:

`cnc`: Statistical information related to CONNECT/DISCONNECT.

In the case of a HiRDB/Parallel Server, the output destination file is the system log file at a front-end server.

2.3.44 Operands related to a client group

198) pdcltgrp -g *client-group-name -u guaranteed-number-of-connected-users-per-group*

This operand is specified when the connection frame guarantee facility for a client group is used. For details about the connection frame guarantee facility for a group, see the *HiRDB Version 8 System Operation Guide.*

-g *client-group-name* ~ <alphabetics>((1-2 characters))

Specifies the name of a client group.

Client type	Client group name	Remarks
X/Open XA interface	XA[1]	Client that accesses HiRDB via the X/Open XA interface. Even if the client is a PC or WS, the client group is XA as long as the X/Open XA interface is used for accessing HiRDB.
Distributed client	DF	Client that uses the distributed database facility to access HiRDB from another node. Even if the client is a PC or WS, the client group is DF as long as the distributed database facility is used for accessing HiRDB.
PC client	PC[2]	Windows and Linux clients
WS client	WS[2]	UNIX client
Mainframe client	MF	VOS3 client
User-defined client group[3]	One uppercase character[1,4]	—

Legend:

—: Not applicable.

[1] To specify this client group, the HiRDB client version must be one of the following:

- 04-05 (cannot be specified for 05-00 or 05-01)
- 05-02 or later

[2] To specify this client group, the HiRDB client version must be the following:

- 04-00 or later

[3] Up to 10 user-defined groups can be specified.

[4] The client group name specified here is specified for the PDCLTGRP operand of the client environment definition. If the values specified for the pdcltgrp and PDCLTGRP operands do not match, the value specified here is ignored.

-u *guaranteed-number-of-connected-users-per-group*

~ <unsigned integer>((1-1999))

Specifies the number of connected users for the client group.

Even when accesses to HiRDB from other client groups or utilities are concentrated, the users (up to the number specified here) from the applicable client group are guaranteed access to HiRDB.

Notes

- The total number of users specified by the -u option must not exceed the value specified in the pd_max_users operand; otherwise, HiRDB will not start.
- This facility is applicable only to the database definition utility.

2.3.45 Operands related to plug-ins

199) pdplugin -n *plug-in-name*

~ <identifier>((1-30 characters))

Specifies the name of a plug-in to be used in HiRDB. This operand is omitted when no plug-ins are to be used.

For the names of the plug-ins that can be specified here, see the manuals for the respective plug-ins.

Conditions

- A plug-in specified here must have been registered into HiRDB in advance with the pdplgrgst command.

2.3.46 Operands related to Real Time SAN Replication

For details about Real Time SAN Replication, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide*.

200) HORCMINST *RAID-Manager-instance-number*

~ <unsigned integer>

Specifies the instance number of the RAID Manager that defines the pair logical volume to be used by HiRDB when Real Time SAN Replication is used. For details about the range of values that can be specified, see the manual *RAID Manager*.

2.3.47 Operands related to shared memory

201) SHMMAX *maximum-shared-memory-segment-siz*e

~ **<unsigned integer> (MB)**

- 32-bit mode: **((6-2047)) <<200>>**
- 64-bit mode: **((6-4194304)) <<1024>>**

Specifies, in megabytes, the maximum segment size for the shared memory for global buffer.

Specification guidelines

- For this operand, specify a value that is equal to or less than the value specified for the OS's operating system parameter `shmmax` (`shminfo_shmmax` for Solaris, and `SHMMAX` for Linux). If you change the `shmmax` value, also change the value of this operand. For the method of estimating operating system parameters, see the *HiRDB Version 8 Installation and Design Guide*.
- HiRDB allocates shared memory segments for global buffers up to the size specified for this operand. If the total size of the global buffers allocated to RDAREAs inside the server machine exceeds the value specified for this operand, multiple shared memory segments are allocated. If the global buffers are assigned to multiple shared memory segments, the overhead for accessing the shared memory segments increases, causing throughput to decrease. Therefore, try to estimate for this operand a value that enables all of the global buffers to fit inside a single shared memory segment.

  You can use the `pdls -d mem` command to check whether all of the global buffers have been assigned to a single shared memory segment. When there is one server per unit, two pieces of shared memory segment information are displayed. Therefore, when there are multiple servers per unit, all of the global buffers have been assigned to a single shared memory segment if *segment-information-count* - 1 = *server-count* is displayed.
- Allocate at least two shared memory segments (for the unit controller, server, and global buffers) to each unit. You can allocate up to16 shared memory segments.
- When global buffers are dynamically changed, new shared memory

segments are allocated and the dynamically changed global buffers are assigned to these new shared memory segments. Use the `pd_max_add_dbbuff_shm_no` operand to specify the maximum number of shared memory segments that can be allocated.

Notes

- When the following condition is satisfied, the required shared memory cannot be allocated, and thus HiRDB cannot be started:

  SHMMAX-value > shmmax-value

- The number of shared memory segments is restricted by the OS's operating system parameters `shmmni` and `shmseg` (`shminfo_shmmin` and `shminfo_shmseg` for Solaris, and `SHMMIN` and `SHMSEG` for Linux). Consequently, when you are allocating multiple shared memory segments for global buffers, you may not be able to allocate shared memory beyond these restrictions. In this case, take one of the following actions:

  - Reduce the number of global buffer sectors.
  - Increase the values of `shmmni` and `shmseg`.
  - Delete unnecessary global buffers.

For details on the shmmax, shmmni, and shmseg parameters, see the applicable OS manual.

Relationship to other operands

This operand is related to the following operands.

- `pd_dbbuff_modify`
- `pdbuffer`
- `pd_max_add_dbbuff_no`
- `pd_max_add_dbbuff_shm_no`

2.3.48 Operands related to date and time

202) TZ *time-zone*

~ <character string> <<JST-9>>

Specifies the time zone to be applied to the date and time of the log to be output. Time zone means the environment variable in which the environment for displaying the time is specified.

2.3.49 Operands related to the message output suppression facility

203) pdmlgput -s *output-selection*

{-c ALL |

-m *message-ID*[,*message-ID*]...}

Controls whether to allow HiRDB to output messages in `syslogfile`.

-s *output-selection*

Specifies whether to suppress message output.

`Y`: Outputs messages.

`N`: Suppresses message output.

-c ALL

Specify this option to suppress all messages that HiRDB output in `syslogfile`.

When specifying this option, specify `N` for the `-s` option.

-m *message-ID*[,*message-ID*]...

Specifies the IDs of the messages that are to be output or that are to be suppressed. Do not enter the message importance that follows the hyphen (-) in a message ID. (For example, for `KFPS01820-E`, specify `KFPS01820`.)

Specifying a message that is not designed to be output in `syslogfile` for this option is the same as not specifying it.

Operand rules

- You can specify multiple lines of this operand.
- If multiple lines of control are specified for the same message using the `ALL` specification or message ID specification, the latter specification takes precedence.

Relationship to other operands

- `pd_mlg_file_size`: Specifies the maximum size of the message log file.
- `pd_mlg_msg_log_unit`: Specifies the output destination (the system manager unit or the unit that outputs the message) for message logs.

2.3.50 Operands related to character encoding

204) pd_substr_length = 3 | 4 | 5 | 6

Specifies the maximum number of bytes used to represent a single character. This operand is applicable when Unicode (UTF-8) is specified as the character encoding and affects the length of the results of the `SUBSTR` scalar function.

For details about the `SUBSTR` scalar function, see the manual *HiRDB Version 8*

SQL Reference.

Specification guidelines

If you use only characters in the range of UCS-2 (1 to 3 bytes), you can omit this operand. To use characters in the range of UCS-4 (1 to 6 bytes), you must specify the maximum number of bytes used to represent a single character.

If the specified value is greater than the actual number of bytes used to represent a single character, the result of the `SUBSTR` scalar function may become long. Therefore, evaluate the value to be specified in this operand on the basis of the character encoding actually used.

Notes

- According to the ISO/IEC Standard 10646, a range of 1 to 4 bytes is assigned per character and bytes 5 and 6 are reserved for future standards. Although HiRDB supports a range of 1 to 6 bytes per character, problems may be introduced in the future if you use a range of 5 or 6 bytes but no characters have been assigned using that number of bytes.
- In Unicode (UTF-8) supported by version 06-02, you can use a maximum of 3 bytes per character. Version 08-00 supports a maximum of 6 bytes. Therefore, with some data, the number of bytes representing a single character may increase.
- If all the following conditions are satisfied, you must re-create SQL objects in routines and re-define view tables:

  - Unicode (UTF-8) is specified as the character encoding.

  - A value other than 3 is to be specified in the new `pd_substr_length` operand; or, in an environment where the `pd_substr_length` operand has already been specified, routines and view tables are to be defined and the `pd_substr_length` operand value is to be changed.

  - A `SUBSTR` scalar function specifying the mixed character string type (`MCHAR` or `MVARCHAR`) as its argument is to be specified in routines and view definitions.

  You can specify the `PDSUBSTRLEN` operand in the client environment definition to control each connection, or you can specify the `SUBSTR LENGTH` SQL compile option to control each routine.
- The tables below show the priority among the `pd_substr_length` operand and other operands when Unicode (UTF-8) is specified as the character encoding.

  When control is performed for each routine

Control timing	pd_substr_length operand	Client environment definition PDSUBSTRLEN	SQL compile option SUBSTR LENGTH	Remarks
During definition	Valid (2)	Invalid	Valid (1)	Default value `pd_substr_length`
During call or function call	Invalid (depends on the specification made during definition)	Invalid (depends on the specification made during definition)	—	None

Legend:

( ): Priority level

—: Cannot be specified

When control is performed for each view table

Control timing	pd_substr_length operand	Client environment definition PDSUBSTRLEN	SQL compile option SUBSTR LENGTH	Remarks
During definition	Valid (2)	Valid (1)	—	Default value `pd_substr_length`
During operation	Invalid (depends on the specification made during definition)	Invalid (depends on the specification made during definition)	—	None

Legend:

( ): Priority level

—: Cannot be specified

Chapter

3. Unit Control Information Definition

This chapter explains the operands of the unit control information definition.

This chapter contains the following sections:

3.1 Operand formats

A unit control information definition defines information about a unit. This section explains the formats used to specify the operands of a unit control information definition. Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *3.2 Operands whose specification values can be changed* and *3.3 Operand explanations*.

For users who are creating HiRDB system definitions for the first time

The first step is to determine the values to be specified for the operands shown in boldface type. In principle, HiRDB can be started once the boldface operands have been specified.

For users of the standby-less system switchover (effects distributed) facility

When you use the standby-less system switchover (effects distributed) facility, you can specify only certain operands. For details, see *I. List of Operands That Can Be Specified When Using the Standby-less System Switchover (Effects Distributed) Facility (Unit Control Information Definition)*.

No.	Format	Operand category
1	**set pd_unit_id =** ***unit-identifier***	System structure
2	[set pd_hostname = *host-name*]	
3	[set pd_max_server_process = *maximum-number-of-concurrently-activated-server-processes*]*	Maximum concurrent executions
4	[set pd_term_watch_count = *maximum-number-of-successive-abnormal- terminations*]*	HiRDB startup
5	[set pd_server_entry_queue = spnfifo | fifo | loop]*	HiRDB processing
6	[set pd_thdlock_wakeup_lock = Y | N]*	
7	[set pd_thdlock_pipe_retry_interval = pd_thdlock_pipe_retry_interval = *thread-lock-release-check-interval*]*	
8	[set pd_thdlock_retry_time = *thread-lock-sleep-time*]*	

No.	Format	Operand category
9	[set pd_thdspnlk_spn_count = *thread-spin-lock-spin-count*]*	
10	[set pd_db_io_error_action = dbhold | unitdown]*	
11	[set pd_max_recover_process = *concurrently-executable-full-recovery-processes-count*]*	Full recovery processing
12	[set pd_watch_time = *SQL-maximum-execution-time*]*	System monitoring
13	[set pd_down_watch_proc = *upper-limit-for-server-process-abnormal-terminations* [, *monitoring-interval*]]*	
14	[set pd_cwaittime_report_dir = *SQL-runtime-warning-information-file-output-destination-directory*]*	SQL runtime warning output facility
15	[set pd_cwaittime_report_size = *SQL-runtime-warning-information-file-maximum-size*]*	
16	[set pd_uap_exerror_log_dir = *SQL-error-report-file-storage-directory*]	Facility for output of extended SQL error information
17	[set pd_uap_exerror_log_size = *SQL-error-report-file-maximum-size*]	
18	[set pd_uap_exerror_log_param_size = *maximum-data-size-of-parameter-information-to-be-output-to-error-log-file-and-SQL-error-report-file*]	
19	[set pd_lck_wait_timeout = *lock-release-wait-time*]*	Lock
20	[set pd_lck_release_detect = interval | pipe]*	
21	[set pd_lck_release_detect_interval = *lock-release-detection-interval*]*	
22	[set pd_lck_deadlock_info = Y | N]*	

No.	Format	Operand category
23	`[set pd_shmpool_attribute = free	fixed]`*	Shared memory
24	`[set pd_dbbuff_attribute = free	fixed]`*	
25	`[set pd_stj_file_size =` *maximum-statistics-log-file-size*`]`*	Statistical information
26	`[set pd_stj_buff_size =` *statistics-log-buffer-size*`]`*	
27	`[set pd_rpc_trace = Y	N]`*	RPC trace information
28	`[set pd_rpc_trace_name = "`*name-of-RPC-trace-collection-file*`"]`*	
29	`[set pd_rpc_trace_size =` *RPC-trace-collection-file-size*`]`*	
30	`[set pd_cancel_dump = put	noput]`*	Troubleshooting information
31	`[set pd_dump_suppress_watch_time` =*troubleshooting-information-output-suppression-time*`]`*	
32	`[set pd_spool_cleanup_interval =` *troubleshooting-information-deletion-interval*`]`*	
33	`[set pd_spool_cleanup_interval_level =` *number-of-days*`[,`*deletion-type*`]]`*	
34	`[set pd_spool_cleanup = normal	force	no]`*	
35	`[set pd_spool_cleanup_level =` *number-of-days* [,*deletion-type*`]]`*	
36	`[set pd_module_trace_max =` *maximum-number-of-module-traces-that-can-be-stored*`]`*	
37	`[set pd_module_trace_timer_level =` *module-trace-output-time-acquisition-method*`]`	
38	`[set pd_dbbuff_wait_interval` = *global-buffer-occupation-state-check-interval*`]`*	Global buffer

No.	Format	Operand category
39	[set pd_dbbuff_wait_spn_count = *maximum-spin-loop-count-for-global-buffer-occupation-state-checking*]*	
40	**set pd_syssts_file_name_1 = "*logical-file-name*" , "*file-a-status-file-name*" , "*file-b-status-file-name*"**	Unit status files
	[set pd_syssts_file_name_2 = "*logical-file-name*" , "*file-a-status-file-name*" , "*file-b-status-file-name*"]	
	[set pd_syssts_file_name_3 = "*logical-file-name*" , "*file-a-status-file-name*" , "*file-b-status-file-name*"]	
	[set pd_syssts_file_name_4 = "*logical-file-name*" , "*file-a-status-file-name*" , "*file-b-status-file-name*"]	
	[set pd_syssts_file_name_5 = "*logical-file-name*" , "*file-a-status-file-name*" , "*file-b-status-file-name*"]	
	[set pd_syssts_file_name_6 = "*logical-file-name*" , "*file-a-status-file-name*" , "*file-b-status-file-name*"]	
	[set pd_syssts_file_name_7 = "*logical-file-name*" , "*file-a-status-file-name*" , "*file-b-status-fil*e-name"]	
41	[set pd_syssts_subfile_name_1 = "*logical-file-name*" , "*primary-status-file-name-for-log-application-processing*" , "*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_syssts_subfile_name_2 = "*logical-file-name*" , "*primary-status-file-name-for-log-application-processing*" , "*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_syssts_subfile_name_3 = "*logical-file-name*" , "*primary-status-file-name-for-log-application-processing*" , "*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_syssts_subfile_name_4 = "*logical-file-name*" , "*primary-status-file-name-for-log-application-processing*" , "*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_syssts_subfile_name_5 = "*logical-file-name*" , "*primary-status-file-name-for-log-application-processing*" , "*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_syssts_subfile_name_6 = "*logical-file-name*" , "*primary-status-file-name-for-log-application-processing*" , "*secondary-status-file-name-for-log-application-processing*"]	

No.	Format	Operand category
	[set pd_syssts_subfile_name_7 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
42	[set pd_syssts_initial_error = stop | continue | excontinue]	Unit status files (when an error occurs)
43	[set pd_syssts_singleoperation = stop | continue]	
44	[set pd_syssts_last_active_file = "*logical-file-name*"]	
45	[set pd_syssts_last_active_side = A | B]	
46	[set pd_syssts_last_active_subfile = "*logical-file–name*"]	
47	[set pd_syssts_last_active_side_sub = A | B]	
48	[set pd_audit = Y | N]*	Security audit facility
49	[set pd_aud_file_name = *HiRDB-file-system-area-name-for-audit-trail-file*]*	
50	[set pd_aud_max_generation_size = *audit-trail-file-maximum-size*]*	
51	[set pd_aud_max_generation_num = *maximum-audit-trail-file-count*]*	
52	[set pd_aud_async_buff_size = *size-of-buffer-used-for-asynchronous-output-of-audit-trail-file*]*	
53	[set pd_aud_async_buff_count = *number-of-buffer-sectors-used-for-asynchronous-output-of-audit-trail-file*]*	
54	[set pd_aud_async_buff_retry_intvl = *retry-interval-for-allocation-of-a-buffer-to-be-used-for-asynchronous-output-of-audit-trail-file*]*	
55	[set pd_ha_acttype = monitor | server]	System switchover facility
56	[set pd_ha_unit = nouse]	

No.	Format	Operand category
57	[set pd_ha_restart_failure = *command-name*]	
58	[set pd_ha_switch_timeout = Y | N]*	
59	[set pd_ha_server_process_standby = Y | N]	
60	[set pd_ha_agent = standbyunit | server | activeunits]	
61	[set pd_ha_max_act_guest_servers = *maximum-number-of-acceptable-guest-BES*]	
62	[set pd_ha_max_server_process = *maximum-number-of-user-server-processes-inside-accepting-unit*]	
63	[set pd_ha_process_count = *number-of-processes-resident-inside-unit-after-acceptance-of-guest-BES*]	
64	[set pd_ha_resource_act_wait_time = *maximum-wait-time-for-resource-activation*]*	
65	[set pd_ha_ipaddr_inherit = Y | N]*	
66	[set pd_rpl_hdepath = *extracted-side-HiRDB-Datareplicator-directory-name*]	HiRDB Datareplicator
67	[set pd_service_port = *port-number-for-high-speed-connection-from-client*]*	Communication processing
68	[set pd_change_clt_ipaddr = 0 | 1]*	
69	[set pd_registered_port = "*port-number-reservation-range*" *[,"port-number-reservation-range"]...*]*	
70	[set pd_registered_port_check = Y | N | C | W]*	
71	[set pd_registered_port_level = 0 | 1]*	
72	[set pd_ipc_send_retrycount = *process-to-process-send-retries-count*]*	

No.	Format	Operand category
73	[set pd_ipc_send_retrysleeptime = *process-to-process-send-retry-sleep-time*]*	
74	[set pd_ipc_send_count = *server-to-server-send-retries-count*]*	
75	[set pd_ipc_recv_count = *server-to-server-receive-retries-count*]*	
76	[set pd_ipc_inet_bufsize = *send-receive-buffer-size-for-server-unit-to-unit-communication*]*	
77	[set pd_ipc_unix_bufsize = *send-receive-buffer-size-for-communication-within-server-unit*]*	
78	[set pd_tcp_inet_bufsize = *send-receive-buffer-size-for-communication-with-HiRDB-client-outside-host-where-HiRDB-server-resides*]*	
79	[set pd_tcp_unix_bufsize = *send-receive-buffer-size-for-communication-with-HiRDB-client-inside-host-where-HiRDB-server-resides*]*	
80	[set pd_java_archive_directory = *"JAR-file-storage-directory"*]*	Java
81	[set pd_java_classpath = *"Java-class-path"*]*	
82	[set pd_java_runtimepath = *"Java-Runtime-Environment-root-directory"*]*	
83	[set pd_java_libpath = *"Java-virtual-machine-library-directory"*]*	
84	[set pd_java_stdout_file = *"Java-virtual-machine-standard-output-and-standard-error-output-destination-file"*]*	
85	**[putenv SHMMAX *maximum-shared-memory-segment-size*]***	Shared memory
86	**[putenv PDCONFPATH *shared-directory-name*]**	Sharing of HiRDB system definition files

* When this operand is omitted, the value specified for the same operand in the system

common definition is used.

3.2 Operands whose specification values can be changed

The values of some of the unit control information definition operands can be changed in the individual server definitions. These operands are indicated below. After a planned termination, forced termination, or abnormal termination of HiRDB, some HiRDB system definition operands can be modified while others cannot be modified. The operands that can be modified are indicated below.

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
1	pd_unit_id	—	Y	—	—	—	—	—	N	N
2	pd_hostname	—	Y	—	—	—	—	—	N	N
3	pd_max_server_process	Y	Y	—	—	—	—	—	Y	Y
4	pd_term_watch_count	Y	Y	—	—	—	—	—	Y	Y
5	pd_server_entry_queue	Y	Y	—	—	—	—	—	Y	Y
6	pd_thdlock_wakeup_lock	Y	Y	—	—	—	—	—	Y	Y
7	pd_thdlock_pipe_retry_interval	Y	Y	—	—	—	—	—	Y	Y
8	pd_thdlock_retry_time	Y	Y	—	—	—	—	—	Y	Y
9	pd_thdspnlk_spn_count	Y	Y	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
10	pd_db_io_error_action	Y	Y	—	—	—	—	—	Y	Y
11	pd_max_recover_process	Y	Y	—	—	—	—	—	Y	Y
12	pd_watch_time	Y	Y	—	—	—	—	—	Y	Y
13	pd_down_watch_proc	Y	Y	—	—	—	—	—	Y	Y
14	pd_cwaittime_report_dir	Y	Y	—	—	—	—	—	Y	Y
15	pd_cwaittime_report_size	Y	Y	—	—	—	—	—	Y	Y
16	pd_uap_exerror_log_dir	Y	Y	—	—	—	—	—	Y	Y
17	pd_uap_exerror_log_size	Y	Y	—	—	—	—	—	Y	Y
18	pd_uap_exerror_log_param_size[4]	Y	Y	—	—	—	—	—	Y	Y
19	pd_lck_wait_timeout	Y	Y	—	—	—	—	—	Y	Y
20	pd_lck_release_detect	Y	Y	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
21	`pd_lck_release_detect_interval`	Y	Y	—	—	—	—	—	Y	Y
22	`pd_lck_deadlock_info`	Y	Y	—	—	—	—	—	Y	Y
23	`pd_shmpool_attribute`	Y	Y	—	—	—	—	—	Y	Y
24	`pd_dbbuff_attribute`	Y	Y	—	—	—	—	—	Y	Y
25	`pd_stj_file_size`	Y	Y	—	—	—	—	—	Y[3]	Y
26	`pd_stj_buff_size`	Y	Y	—	—	—	—	—	Y[3]	Y
27	`pd_rpc_trace`	Y	Y	Y	Y	Y	Y	Y	Y	Y
28	`pd_rpc_trace_name`	Y	Y	Y	Y	Y	Y	Y	Y	Y
29	`pd_rpc_trace_size`	Y	Y	Y	Y	Y	Y	Y	Y	Y
30	`pd_cancel_dump`	Y	Y	—	—	—	—	—	Y	Y
31	`pd_dump_suppress_watch_time`	Y	Y	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
32	`pd_spool_cleanup_interval`	Y	Y	—	—	—	—	—	Y	Y
33	`pd_spool_cleanup_interval_level`	Y	Y	—	—	—	—	—	Y	Y
34	`pd_spool_cleanup`	Y	Y	—	—	—	—	—	Y	Y
35	`pd_spool_cleanup_level`	Y	Y	—	—	—	—	—	Y	Y
36	`pd_module_trace_max`	Y	Y	Y	Y	Y	Y	Y	Y	Y
37	`pd_module_trace_timer_level`	Y	Y	Y	Y	Y	Y	Y	Y	Y
38	`pd_dbbuff_wait_interval`	Y	Y	—	—	—	—	—	Y	Y
39	`pd_dbbuff_wait_spn_count`	Y	Y	—	—	—	—	—	Y	Y
40	`pd_syssts_file_name_1` — 7	—	Y	—	—	—	—	—	Y[1]	Y[1]
41	`pd_syssts_subfile_name_1` to 7	—	Y	—	—	—	—	—	N	N

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
42	pd_sysssts_initial_error	—	Y	—	—	—	—	—	Y	Y
43	pd_sysssts_singleoperation	—	Y	—	—	—	—	—	Y	Y
44	pd_sysssts_last_active_file	—	Y	—	—	—	—	—	Y	Y
45	pd_sysssts_last_active_side	—	Y	—	—	—	—	—	Y	Y
46	pd_sysssts_last_active_subfile	—	Y	—	—	—	—	—	Y	Y
47	pd_sysssts_last_active_side_sub	—	Y	—	—	—	—	—	Y	Y
48	pd_audit	Y	Y	—	—	—	—	—	N	N
49	pd_aud_file_name	Y	Y	—	—	—	—	—	Y	Y
50	pd_aud_max_generation_size	Y	Y	—	—	—	—	—	Y	Y
51	pd_aud_max_generation_num	Y	Y	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
52	pd_aud_async_buff_size	Y	Y	—	—	—	—	—	Y	Y
53	pd_aud_async_buff_count	Y	Y	—	—	—	—	—	Y	Y
54	pd_aud_async_buff_retry_intvl	Y	Y	—	—	—	—	—	Y	Y
55	pd_ha_acttype	—	Y	—	—	—	—	—	N	N
56	pd_ha_unit	—	Y	—	—	—	—	—	N	N
57	pd_ha_restart_failure	—	Y	—	—	—	—	—	Y	Y
58	pd_ha_switch_timeout	Y	Y	—	—	—	—	—	Y	Y
59	pd_ha_server_process_standby	—	Y	—	—	—	—	—	N	N
60	pd_ha_agent	—	Y	—	—	—	—	—	N	N
61	pd_ha_max_act_guest_servers	—	Y	—	—	—	—	—	N	N
62	pd_ha_max_server_process	—	Y	—	—	—	—	—	N	Y^2

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
63	pd_ha_process_count	—	Y	—	—	—	—	—	Y	Y
64	pd_ha_resource_act_wait_time	Y	Y	—	—	—	—	—	Y	Y
65	pd_ha_ipaddr_inherit	Y	Y	—	—	—	—	—	N	N
66	pd_rpl_hdepath	—	Y	—	—	—	—	—	N	N
67	pd_service_port	Y	Y	—	—	—	—	—	Y	Y
68	pd_change_clt_ipaddr	Y	Y	—	—	—	—	—	Y	Y
69	pd_registered_port	Y	Y	—	—	—	—	—	N	Y
70	pd_registered_port_check	Y	Y	—	—	—	—	—	N	Y
71	pd_registered_port_level	Y	Y	—	—	—	—	—	N	Y
72	pd_ipc_send_retrycount	Y	Y	—	—	—	—	—	Y	Y
73	pd_ipc_send_retrysleeptime	Y	Y	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
74	`pd_ipc_send_count`	Y	Y	—	—	—	—	—	Y	Y
75	`pd_ipc_recv_count`	Y	Y	—	—	—	—	—	Y	Y
76	`pd_ipc_inet_bufsize`	Y	Y	—	—	—	—	—	Y	Y
77	`pd_ipc_unix_bufsize`	Y	Y	—	—	—	—	—	Y	Y
78	`pd_tcp_inet_bufsize`	Y	Y	—	—	—	—	—	Y	Y
79	`pd_tcp_unix_bufsize`	Y	Y	—	—	—	—	—	Y	Y
80	`pd_java_archive_directory`	Y	Y	—	—	—	—	—	Y	Y
81	`pd_java_classpath`	Y	Y	—	—	—	—	—	Y	Y
82	`pd_java_runtimepath`	Y	Y	—	—	—	—	—	Y	Y
83	`pd_java_libpath`	Y	Y	—	—	—	—	—	Y	Y
84	`pd_java_stdout_file`	Y	Y	Y	Y	Y	Y	Y	Y	Y
85	`SHMMAX`	Y	Y	—	—	—	—	—	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
86	PDCONFPATH	—	Y	—	—	—	—	—	Y	Y

Y: Yes, specification value can be modified.

N: No, specification valued cannot be modified.

— : Specification value cannot be modified because the operand is not applicable.

SYS: System common definition

UNT: Unit control information definition

SVR: Server common definition

SDS: Single server definition

FES: Front-end server definition

DS: Dictionary server definition

BES: Back-end server definition

[1] Operands may be added, but not deleted or modified.

[2] If the specified value is too small, it may not be possible to restart HiRDB. When this is the case, restore the value to what it was before the change and then restart HiRDB.

[3] If the specified value is too large, it may not be possible to restart HiRDB. When this is the case, restore the value to what it was before the change and then restart HiRDB.

[4] The value of the following HiRDB system definition operand can be changed for each client in its client environment definition. To change values for a client, you must specify the applicable operand in its client environment definition. For details about the client environment definition, see the *HiRDB Version 8 UAP Development Guide*.

HiRDB system definition operand	Client environment definition operand
pd_uap_exerror_log_param_size	PDUAPEXERLOGPRMSZ

3.3 Operand explanations

3.3.1 Operands related to system structure

1) pd_unit_id = ***unit-identifier***

~ <identifier>((4 characters))

This operand is required.

Specifies an identifier for the unit. The unit identifier specified here must have been specified in the `pdunit -u` operand in the system common definition.

If multiple HiRDB/Single Servers are used, specify a unique identifier for each system.

Each utility special unit must also have a unique unit identifier. If you specify a unit identifier that has already been specified, an error will occur when the utility special unit is started.

For a HiRDB/Parallel Server, the specified identifier must be unique within the system.

Notes

When a unit identifier is changed, the following files must be reinitialized:

- Unit status files
- System log files

2) pd_hostname = ***host-name***

~ <host name>((1-32 characters))

Specifies the standard host name, IP address, or FQDN for the server machine in which the unit was defined.

Specification guidelines

- The host name specified here must be defined under `/etc/brc.ipc`. The `hostname` command or the `gethostname` function can be used to obtain the host name.
- You cannot omit this operand if you use Real Time SAN Replication. Specify the standard host name of the main site for the `pd_hostname` operand of the main site; specify the standard host name of the remote site for the `pd_hostname` operand of the remote site.
- If the system switchover facility that inherits IP addresses is used and this operand specifies a loopback address (applicable to a HiRDB/

Single Server) or a host name other than the standard host, specify the host name with a relocatable IP address in the -m option of the pdunit operand. Also specify the same host name in the -n option of the pdunit operand.

- For details about the specification value, see Table *2-2 Host name when pdunit operand's -x option and pd_hostname operand are specified (for HiRDB/Single Server)* or Table *2-3 Host name when pdunit operand's -x option and pd_hostname operand are specified (for HiRDB/Parallel Server)* in *2.3.39 Operands related to unit structure*.

3.3.2 Operands related to maximum concurrent executions

3) pd_max_server_process = ***maximum-number-of-concurrently-activated-server-processes***

~ <unsigned integer>((50-10000))

Specifies the maximum number of processes that can be activated at the same time in the unit. The number of server processes includes the number of processes for the system server, individual servers, utilities, etc. (the system server is a server that is used internally by HiRDB).

Specification guidelines

See the explanation of the pd_max_server_process operand of the system common definition.

Relationships to other operands

This operand's value limits the maximum value of the pd_process_count operand.

Notes

- This specification value includes the number of processes for servers and utilities in the unit. If this value is too small, the following may occur:
  - Unit or server startup process results in an error
  - Transaction recovery cannot be performed
  - HiRDB planned termination cannot be performed
- Because the number of processes that can actually be activated depends on factors such as the system resources, it may be necessary to adjust the resources or to change the locations of servers in some cases.

3.3.3 Operands related to HiRDB startup

4) pd_term_watch_count =

maximum-number-of-successive-abnormal-terminations

~ <unsigned integer>((1-3))

Specifies the maximum number of times in a row that abnormal termination can occur. If HiRDB terminates abnormally during HiRDB (or unit) restart attempts for the number of times in a row specified in this operand, restart processing is canceled. For example, if `2` is specified here, restart processing is canceled after two HiRDB restarts in a row terminate abnormally.

The HiRDB restart attempts can be either automatic or manual, depending on the value specified in the `pd_mode_conf` operand.

When HiRDB terminates abnormally after the number of abnormal terminations specified in this operand has been reached, the `KFPS00715-E` message is output. In this case, remove the cause of the error and then execute the `pdrpause` command in the unit indicated in the `KFPS00715-E` message.

3.3.4 Operands related to HiRDB processing

5) pd_server_entry_queue = spnfifo | fifo | loop

If contention occurs in the HiRDB server process during concurrent execution of UAPs, processing requests may sometimes be temporarily queued. This operand specifies what HiRDB must do in this case. Note that *process contention* in this case means that multiple processes are simultaneously trying to lock internal resources, such as tables and RDAREAs, when transactions are running on the HiRDB server process. Only a single process is allowed to lock internal resources at any point in time. *Spin* referred to in the following explanation means a process for acquiring the right to execute a lock. When another process releases the right to execute a lock, a process that is spinning has a higher probability of acquiring the right to execute a lock.

`spnfifo`:

A processing request that occurs first is given higher priority. However, because the process is spun a certain number of times before being registered in a queue, the priority order is not perfect. This method is used in Version 06-00 and older versions.

`fifo`:

A processing request that occurs first is given higher priority than when `spnfifo` is specified. Because the process is not spun a certain number of times before a process is registered in a queue, the priority order is maintained better than when `spnfifo` is specified. This method also reduces the CPU load.

`loop`:

All processing requests are given the same priority. When processes are registered in a queue, they are spun at high speed. Specifying `loop` may improve the response during concurrent execution of UAPs. However, this method places a greater load on the CPU than other methods.

Specification guidelines

Normally, you need not specify this operand.

Change the specification value if the processing performance during concurrent execution of UAPs does not improve. Doing so may improve the performance.

6) pd_thdlock_wakeup_lock = Y | N

Specifies a thread lock release notification method.

`Y`:

When issuing a thread lock release notification, a new separate lock is temporarily obtained. Because this temporary lock is obtained, reliable transmission of the release notification is guaranteed.

`N`:

When issuing a thread lock release notification, no new separate lock is temporarily obtained. When no temporary lock is obtained, a transaction may occur that has a longer execution time than other transactions (how much longer is determined by the value specified for the `pd_thdlock_pipe_retry_interval` operand).

Specification guidelines

1. The following table shows the effects of multiplexed transaction execution:

Comparison item	pd_thdlock_wakeup_lock operand value	
	Y	N
Difference in transaction execution time	Reduces the difference.	Increases the difference.
Time required for the completion of all transactions	Lengthens	Shortens

2. The following table shows the recommended values of this operand:

Condition	Recommended value
When a new HiRDB is used	Y

Condition		Recommended value
When HiRDB is already being used	When the execution times of all transactions must be the same during multiplexed transaction execution	`Y`
	When some differences in the execution times of all transactions are allowed during multiplexed transaction execution	`N`

Relationship to other operands

If you specify `N` in this operand or you omit this operand, evaluate the specification of the `pd_thdlock_pipe_retry_interval` operand.

7) pd_thdlock_pipe_retry_interval = *thread-lock-release-check-interval*

～ <unsigned integer>((0-2147483647))(microseconds)

Specifies in microseconds the interval at which to check for thread lock release. Specify this operand if `pd_thdlock_wakeup_lock = N` (default value) is specified. In all other cases, there is no need to specify this operand.

If multiple transactions are executed concurrently, differences occur in their execution times. The value specified for this operand provides an estimate for the maximum execution time difference (this value is not exact).

Specification guidelines

- If this operand is to be specified, specify `1000` for it.
- If the specification of `1000` increases the CPU usage rate too much, specify a greater value.
- Even if you specify a value less than `1000`, performance does not improve over a case in which `1000` is specified.

Note

If `0` is specified for this operand, the release check is repeated within short intervals and may result in an extremely high CPU usage rate.

8) pd_thdlock_retry_time = thread-lock-sleep-time

～ <unsigned integer>((1-1000000))(microseconds)

Specifies in microseconds the thread lock sleep time. If this operand is specified when all the conditions listed below are satisfied, the CPU usage rate may decrease. Otherwise, there is no need to specify this operand.

- The CPU usage rate is very high.
- Reducing the CPU usage rate is necessary even if it results in a reduction in performance.

- 0 is specified for the pd_thdlock_sleep_func operand.

The following describes the HiRDB processing based on the combination of the pd_thdlock_sleep_func and the pd_thdlock_retry_time operand values:

pd_thdlock_sleep_func operand value	pd_thdlock_retry_time operand value	
	1 to 10000	10001 to 1000000
0	Each process stands by for the thread lock sleep time specified by select() or Sleep().	
1	The OS determines process allocation using sched_yield() or SwitchToThread() (the pd_thdlock_retry_time operand value is ignored).	Each process stands by for the thread lock sleep time specified by select() or Sleep().

Specification guidelines

- If this operand is to be specified, begin specify 10000 for it.
- If the specification of 10000 increases the CPU usage rate too much, specify a greater value.
- If there is an ample margin in the CPU usage rate, specify a value less than 10000.

Notes

- Reducing the value may not change the performance.
- Specifying 1000 or a greater value may have an adverse effect on performance.

9) pd_thdspnlk_spn_count = ***thread-spin-lock-spin-count***

~ <unsigned integer>((0-2147483647))

Specifies a spin count for thread spin lock. Specifying this operand when all of the following conditions are satisfied may improve the system performance. Otherwise, there is no need to specify this operand.

- An ample margin exists in the CPU usage rate.
- You want to improve performance even if the CPU usage rate must be increased.

Specification guidelines

- If this operand is to be specified, specify a value that is greater than the default value (512).

- Because the specification value depends on the OS type, the processor type, the machine performance, the disk performance, the UAP content, and the number of UAPs concurrently being executed, there is no clear guideline. Determine an appropriate value by varying the specification value and measuring the performance.

Notes

- If the value of this operand is too large, the CPU usage rate may increase, causing problems such as slower OS operation. In this case, decrease the operand value.
- Increasing the value of this operand may not always improve performance.

10) pd_db_io_error_action = dbhold | unitdown

Specifies the processing to be performed by HiRDB when an input/output error occurs in an RDAREA (excluding the master directory RDAREA). If an input/output error occurs in the master directory RDAREA, HiRDB (a unit for a HiRDB/Parallel Server) is always abnormally terminated regardless of the specification in this operand. For the actions to be taken when an RDAREA input/output error occurs, see the *HiRDB Version 8 System Operation Guide*.

An input/output error in this case refers to an error that occurs when a file manipulation attempt by HiRDB fails due to a cause that cannot be determined by HiRDB. When such an error occurs, `-1544` is output as the error code returned in response to a HiRDB file system access request.

`dbhold`:

When an input/output error occurs in an RDAREA, the RDAREA is placed in an error shutdown state.

`unitdown`:

If an input/output error occurs in an RDAREA, HiRDB (a unit for a HiRDB/Parallel Server) is abnormally terminated. However, if an input/output error occurs again following abnormal termination, the RDAREA is placed in an error shutdown state. To enable the specification of `unitdown` again, take one of the following actions:

- Start HiRDB normally.
- Execute the system reconfiguration command (`pdchgconf` command).

Specification guidelines

To determine the specification value for this operand, see *Actions to be taken when an RDAREA input/output error occurs* in the *HiRDB Version 8 System Operation Guide*.

Notes

- If an input/output error occurs when `unitdown` is specified, HiRDB is abnormally terminated. Consequently, if a UAP or utility is being executed in the pre-update log acquisition mode or no-log mode, the processing target RDAREA may go into an error shutdown state.
- If an input/output error occurs during the startup or termination process, HiRDB is not abnormally terminated, even if `unitdown` is specified.
- If Real Time SAN Replication based on the log-only synchronous method is used and an input/output error occurs during log application processing at the log application site, HiRDB does not terminate abnormally even when `unitdown` is specified.

3.3.5 Operands related to full recovery processing

11) pd_max_recover_process = ***concurrently-executable-full-recovery-processes-count***

~ <unsigned integer>((1-10))

Specifies the number of processes to be recovered during full recovery processing. For a HiRDB/Parallel Server, this operand specifies the number of processes to be recovered per server (dictionary server or back-end server).

Condition

Parallel input/output processing is not performed for regular files.

Specification guidelines

- The value to be specified depends on the file type:

Condition		Specification guideline
Character special files	There are at least 10 character special files (per server) that define RDAREAs	10
	There are fewer than 10 character special files (per server) that define RDAREAs	Number of character special files
Regular file		1

- Increasing the value of this operand increases the input/output concurrency during full recovery processing, and thus can shorten the recovery time. However, because a number of processes equaling *value-of-this-operand* × *server-count* are started, determine a value by taking the aforementioned specification value guideline and HiRDB resources into consideration.

3.3.6 Operands related to system monitoring

12) pd_watch_time = *SQL-maximum-execution-time*

~ <unsigned integer>((0-65535)) (seconds)

This operand is applicable to a HiRDB/Parallel Server.

Specifies a maximum execution time for SQL statements that are executed in a HiRDB server process.

If execution of an SQL is not completed within the specified amount of time, execution of that SQL is terminated.

Advantage

If the HiRDB server does not halt execution of an SQL statement, even though the HiRDB client has canceled SQL execution (by forcibly terminating a client process, for example), the HiRDB server may continue to execute the SQL statement and may lock resources for a long time. Specifying this operand places a limit on such a lock time.

Specification guidelines

Specify the largest value among the following time values:

Time specified by the `PDCWAITTIME` operand of the client environment definition

Time specified by the `pd_lck_wait_timeout` operand

Processing time of the SQL statement with the longest execution time

Notes

- When `0` is specified for this operand, the SQL execution time is not monitored.
- When `0` (no time monitoring) is specified for the `PDCWAITTIME` operand of the client environment definition for an SQL statement, the execution time of that SQL statement is not monitored.
- When a value shorter than the SQL time is specified for this operand, processing may terminate during SQL execution, and an SQL error may be reported to the HiRDB client or abnormal termination may be reported to the HiRDB server.
- For a HiRDB/Single Server, SQL maximum execution time is not monitored, even if this operand is specified. If this operand is specified, the specified value is used as the default value of the `pd_lck_wait_timeout` operand. Therefore, Hitachi recommends that you omit this operand for a HiRDB/Single Server.

13) pd_down_watch_proc = *upper-limit-for-server-process-abnormal-terminations*[*, monitoring-interval*]

This operand is used for monitoring the number of abnormal terminations of a HiRDB server process. Processes to be monitored are those that are abnormally terminated by `PDCWAITTIME` over or aborting.

If abnormal terminations of server processes occur frequently, new services may not be accepted. However, because server process abnormal termination does not cause HiRDB abnormal termination, HiRDB is in an online stopped state in effect. When this operand is specified, you can pull HiRDB out of this state by restarting it.

***upper-limit-for-server-process-abnormal-termination*: ~ <unsigned integer>((0-65535))**

If abnormal terminations of server processes exceed the value specified in this operand, HiRDB (an applicable unit for a HiRDB/Parallel Server) is abnormally terminated. This is called the facility for monitoring abnormal process terminations. For details on this facility, see the *HiRDB Version 8 System Operation Guide*.

For a HiRDB/Single Server, abnormal terminations of single server processes are counted. For a HiRDB/Parallel Server, the total of the abnormal terminations in the front-end servers, back-end servers, and dictionary servers inside the unit is counted.

If `0` is specified, abnormal terminations of server processes are not monitored.

***monitoring-interval*: ~ <unsigned integer>((10-3600)) (units: seconds)**

Specifies the interval (in seconds) for monitoring abnormal terminations of server processes.

For example, if `100` is specified, abnormal terminations of server processes are monitored every 100 seconds.

Advantages

- Restart of HiRDB refreshes memory and resource statuses, improving the processing efficiency.
- If abnormal termination of server processes occurs frequently, HiRDB is abnormally terminated, and thus the system can be switched over immediately.

Notes

- When a server process is abnormally terminated, the `KFPS01820-E` message is output. Although this message is also output when the server process is abnormally terminated by the pdcancel command, this is not counted as an abnormal termination.

- For a mutual system switchover configuration, multiple HiRDBs are activated on the same server machine when system switchover occurs. As a result, the system traffic may increase, causing an adverse effect instead. Therefore, if you specify this operand, Hitachi recommends that you restart HiRDB in the system that was abnormally terminated.

Operand rule

A monitoring interval cannot be specified alone. It must be specified with an upper limit for server process abnormal terminations.

Remarks

- If HiRDB is abnormally terminated by the facility for monitoring abnormal process terminations, the KFPS01821-E and KFPS00729-E messages are output.
- The following table shows the causes of server process abnormal termination and the server processes that are included in the abnormal termination count.

Cause of server process abnormal termination	Inclusion in abnormal termination count			
	Single server process	Front-end server process	Dictionary server process	Back-end server process
PDCWAITTIME operand value of the client environment definition has been exceeded.	Y	Y	N[1]	N[1]
pdcancel command	N	N[2]	N	N
Internal forced termination (HiRDB internally issues SIGKILL and terminates a process)	Y[3]	Y[3]	N[1]	N[1]
Abort	Y	Y	Y	Y
Rollback has occurred in a UAP with XA connection.	Y	Y	N	N
Abnormal termination of process other than those described here	Y	Y	Y	Y

Legend:

Y: Included in abnormal termination count

N: Not included in abnormal termination count

[1] If an error is detected in a transaction branch, the abnormal terminations of the front-end server process that have occurred in the same transaction branch are counted.

[2] If the `pdcancel` command is used to forcibly terminate a back-end server process or dictionary server process, the front-end server process is internally and forcibly terminated. In this case, the abnormal termination of the front-end server process may be counted in some cases.

[3] If an error is detected in a global transaction by an OLTP system, the abnormal terminations of the single server process or front-end server process that has occurred in the same global transaction are counted.

3.3.7 Operands related to SQL runtime warning output facility

14) pd_cwaittime_report_dir = *SQL-runtime-warning-information-file-output-destination-directory*

~ <pathname>

Specifies an absolute path name as the output destination directory for the SQL runtime warning information file. Two SQL runtime warning information files (`pdcwwrn1` and `pdcwwrn2`) are created under the directory specified here.

If this operand is omitted, no SQL runtime warning information file is output. However, the warning message (`KFPA20009-W`) is still output.

For a HiRDB/Parallel Server, the SQL runtime warning information files are output to the server machine containing the front-end server to which the UAP that issued the warning target SQL is connected.

Operand rules

- Specify no more than 255 characters for the path name.

15) pd_cwaittime_report_size = *SQL-runtime-warning-information-file-maximum-size*

~ <unsigned integer>((2048-2147473627)) (Bytes)

Specifies the maximum size for the SQL runtime warning information file. The value specified in this operand indicates the size of a single SQL runtime warning information file. Therefore, be careful about the value you specify for this operand because two SQL runtime warning information files are created. For example, if you specify `10,000`, two files, each with a maximum size of 10,000 bytes, are created under the directory.

Specification guidelines

Use the following formula as a guideline when determining the value to be specified for this operand.

(1280 + *SQL-statement-size* (bytes)) × *number-of-pieces-of-warning-information-to-be-stored-in-file*

If a comment or SQL optimization is specified for the SQL statement, also include the size of the comment and the specified SQL optimization size (bytes) in the SQL statement size.

Remarks

- If the volume of data that is output to an SQL runtime warning information file exceeds the value specified by this operand, the output destination is switched to the other file. The two files are alternately used as this process is repeated. During this process, the oldest information is deleted from the switching destination file.
- If the volume of the SQL runtime warning information that is output at one time exceeds the file size, not all of the SQL runtime warning information is output. Only the information that fits in the file size is output. In this case, the hash mark [#] is added to the end of the SQL runtime warning information.

3.3.8 Operands related to the facility for output of extended SQL error information

16) pd_uap_exerror_log_dir = *SQL-error-report-file-storage-directory*

~ <path name of up to 255 characters>

Specifies an absolute path name for the directory in which to store SQL error report files.

Two SQL error report files are created in the specified directory. Their file names are `pduaperrlog1` and `pduaperrlog2`.

If this operand is omitted, no SQL error information is output in an SQL error report file.

For details about the facility for output of extended SQL error information, see the *HiRDB Version 8 System Operation Guide*.

17) pd_uap_exerror_log_size = *SQL-error-report-file-maximum-size*

~ <unsigned integer>((2048-2147483647))(bytes)

Specifies the maximum size of an SQL error report file. The value specified by this operand applies to each of the two SQL error report files that are to be created. When the volume of data that is output to an SQL error report file exceeds the value specified by this operand, the output destination is switched to the other file. The two files are used alternately as this process is repeated. If the volume of the SQL error information that is output at one time exceeds the value specified by this operand, the first through the [*specified value* -1]-th bytes (up to the 999999th byte if `1000000` is specified for this operand) of SQL error information is output. In this case, the hash mark [#] is added to the end of the SQL error information.

Specification guidelines

Determine the value to be specified for this operand by taking into consideration the volume of SQL error information that should be retained. You can use the following computation formula:

$(A + B) \times \textit{volume-to-be-retained}$

- $A = 1100 + \textit{SQL-statement-size}$ (bytes)

  This is the size of each piece of SQL error information, excluding the parameter information output size. If an SQL statement contains a comment or the description of SQL optimization specification, the size of the comment or SQL optimization specification must also be included in the SQL statement size. For details about comments and SQL optimization specification, see the manual *HiRDB Version 8 SQL Reference*.

- $B = (\uparrow \textit{pd\_uap\_exerror\_log\_param\_size-operand-value} \div 16 \uparrow + 1) \times 89 \times \textit{parameter-count}$

  This is the parameter information output size.

18) pd_uap_exerror_log_param_size = *maximum-data-size-of-parameter-information-to-be-output-to-error-log-file-and-SQL-error-report-file*

～ <unsigned integer>((0-32008))(bytes)

Specifies the maximum data size for the parameter information to be output to an error log file and an SQL error report file.

- When `1` or a value greater than `1` is specified

  Parameter information is output to an error log file and an SQL error report file.

- When `0` is specified

  Parameter information is not output to an error log file or an SQL error report file.

If the parameter information is in the variable-length character string type, BLOB type, or BINARY type, the data size area also is included in the specified value.

If the size of the parameter information to be output to an error log file and an SQL error report file exceeds the value specified for this operand, only the parameter information that fits in the file size is output, and the remainder is discarded.

Relationship to other operands

The value of this operand can be changed for each client. To change the operand for a client, specify the `PDUAPEXERLOGPRMSZ` operand in the client environment definition. If both this operand and the `PDUAPEXERLOGPRMSZ` operand are specified, the `PDUAPEXERLOGPRMSZ` operand in the client environment definition takes precedence.

For details about the `PDUAPEXERLOGPRMSZ` operand, see the *HiRDB Version 8 UAP Development Guide*.

3.3.9 Operands related to lock

19) pd_lck_wait_timeout = *lock-release-wait-time*

~ <unsigned integer>((0-65535)) (seconds)

Specifies in seconds the maximum amount of time to wait for lock release (the maximum amount of time in lock release wait status).This is the elapsed time from when a lock release request is placed in wait status until it is released.

If the wait status is not released within the specified amount of time, the SQL will return an error. When `0` is specified, lock wait time will not be monitored and waiting will continue until the wait status is released.

20) pd_lck_release_detect = interval | pipe

Specifies the method to be used by HiRDB to detect lock release (method of detecting whether or not the process that locked a resource has released that lock).

`interval`:

Determine the lock release status by checking the lock management area at a regular interval.

`pipe`:

Use a pipe file (process-to-process communication pipe) to receive the lock release notice.

Specification guidelines

Following are guidelines for specifying this operand:

Specified value	HiRDB processing	Advantages and application criteria
`interval`	Determines whether the lock has been released by checking the lock management area in the shared memory. HiRDB checks the lock management area at a regular interval; this interval is specified by the `pd_lck_release_detect_interval` operand.	Even if the process that has locked a resource releases the lock, the release will not be detected until the next time the lock management area is checked. Consequently, a UAP that has a fast processing time per transaction may end up waiting for a long time. However, this wait does not place any load on the CPU or open any file. If a small value is specified for the `pd_lck_release_detect_interval` operand, the CPU usage may increase too much, adversely affecting the throughput. Specifying `interval` has the effect of reducing the CPU load when a slow CPU is being used.
`pipe`	Uses a pipe file to determine whether or not the lock has been released. The process that has locked a resource sends a lock release notice to the process that is waiting for release. When the process that has locked the resource releases it, the process that is waiting for the release can detect the release. Because a maximum of one file descriptor is used per process, it is necessary to re-estimate the number of file descriptors.	• Throughput improves if the processing time per transaction does not exceed the value specified by the `pd_lck_release_detect_interval` operand. • CPU workload increases if lock release wait occurs frequently. Additionally, the number of open files is increased because pipe files are opened. • Specify `pipe` when there are system resources (CPU and file descriptors) that can be spared.

Notes

- When `pipe` is specified for the first time after HiRDB has been newly installed, the pipe file is initialized during activation of the unit. Consequently, unit activation completion may take longer than expected.
- When `pipe` has been specified, the pipe file is opened when lock-release wait occurs. Therefore, add to the estimate for the file descriptors for the entire system (nfile in the case of HP-UX) the value determined from the following formula (as applicable):

  HiRDB/Single Server:

  `pd_max_users operand` value + `pd_max_reflect_process_count` value

  HiRDB/Parallel Server:

(a + 3 × *server-count-inside-unit*) × b

a: Total of the values determined for all of the servers inside the unit, computed using the following formulas:

Back-end server: Value of `pd_max_bes_process` + `pd_max_reflect_process_count`

Dictionary server: Value of `pd_max_dic_process` + `pd_max_reflect_process_count`

If the `pd_max_bes_process` or `pd_max_dic_process` operand is omitted, use the value of `pd_max_users`. If a unit contains multiple back-end servers, compute a value for each back-end server.

b: 16 (number of threads waiting for lock release)

- When `pipe` is specified in the Linux version, the pipe file is opened when lock-release wait occurs. In this case, it is necessary to specify the maximum number of open files in the system, including open pipe files, using the method described below.

  Specify the following in `/etc/rc.d/rc.local`:

  /sbin/sysctl -w fs.*file-max* = *maximum-number-of-open-files-in-system*

21) pd_lck_release_detect_interval = *lock-release-detection-interval*

~ <unsigned integer>((1-1000)) (Milliseconds)

Specifies the interval at which the lock management area is to be checked.

- When 49 or less is specified

  The interval will begin at the value specified by this operand and thereafter will be 50 milliseconds.

- When 50 or more is specified

  The interval will begin at 50 milliseconds and thereafter will be the value specified by this operand.

Condition

The `interval` option must be specified for the `pd_lck_release_detect` operand.

Specification guidelines

- As a rule, this operand can be omitted. This operand should be specified (instead of using the default value) only when there is a performance problem.
- If too small a value is specified, the CPU workload will increase if lock

release waits occur frequently.

- If too large a value is specified, the wait time may increase.
- The value to be specified can be determined by referring to WAIT TIME in the statistical information related to system operation from the statistics analysis utility. If the wait time that is output in the statistical information is smaller than the value of this operand, the value of this operand should be reduced.

22) pd_lck_deadlock_info = Y | N

Specifies whether or not deadlock information and timeout information are to be output. These types of information are output to a directory named $PDDIR/spool/pdlckinf. For details about the deadlock and timeout information, see the *HiRDB Version 8 System Operation Guide.*

Y: Output deadlock information and timeout information.

N: Do not output deadlock information or timeout information.

3.3.10 Operands related to shared memory

23) pd_shmpool_attribute = free | fixed

Specifies whether to fix the shared memory used by the HiRDB unit controller in memory. Note that the AIX 5L version of HiRDB(32-bit mode) does not support fixing of the shared memory, and therefore specification of fixed for this operand is invalid.

free:

Do not fix the shared memory in memory. Depending on the size of the real memory, another page of shared memory may be created, affecting performance adversely.

fixed:

Fix the shared memory in memory.

Advantage

Fixing the shared memory to be used by the HiRDB in the memory ("fixed" specified) prevents shared memory paging, thus improving the access performance to the shared memory.

Specification guidelines

Whether or not the shared memory is to be fixed should be based on the computed shared memory size and the real memory size of the server machine. Because there is a limit to the amount of memory that can be fixed for the HiRDB, fixing too much shared memory may cause frequent paging of other types of memory. How much real memory there is and the amount of memory to be occupied by the shared memory pool need to be carefully evaluated.

For the formula for computing the shared memory sizes to be used by the unit controller and individual servers, see the *HiRDB Version 8 Installation and Design Guide*.

Notes

The following notes apply to the specification of `fixed` in the AIX 5L version:

- Set operating system parameters for the OS. If the applicable operating system parameters are not set, this operand is ignored. For details about the operating system parameters to be set, see *Specifying parameters unique to AIX 5L* in the *HiRDB Version 8 Installation and Design Guide*.
- Set the server machine's memory and the `VMM` parameter's `maxpin` so that the following condition is satisfied:

  (*machine's memory size*) x (*maxpin setting*)/100 $\geq$ $a + b^{\#} + c$

  a: Size of shared memory segment for the unit controller

  b: Sum of the sizes of all shared memory segments for global buffers

  c: Sum of the sizes of all shared memory segments used by the OS that fixes shared memory or the additional program products

  #: Add this value when `fixed` is specified in the `pd_dbbuff_attribute` operand.
- You can use an AIX command to determine whether or not shared memory has been fixed in memory; for details, see the AIX 5L manual.

24) pd_dbbuff_attribute = free | fixed

Specifies whether to fix the shared memory used by the global buffers in the real memory. Note that the AIX 5L version of HiRDB(32-bit mode) or the Linux version of HiRDB does not support fixing of the shared memory, and therefore specification of `fixed` for this operand is invalid.

`free` (default value for 64-bit mode):

Do not fix the shared memory to be used by the global buffer in the real memory.

`fixed` (default value for 32-bit mode):

Fix the shared memory to be used by the global buffer in the real memory.

Advantage

Fixing the shared memory to be used by the global buffer in the real memory ("fixed" specified) prevents shared memory paging, thus improving the performance of accesses to the shared memory.

Specification guidelines

- Whether or not the shared memory is to be fixed should be determined based on the computed shared memory size and the real memory size of the server machine. If a memory page that is quite large relative to the real memory is fixed, the result may be frequent paging or a virtual memory shortage. Therefore, how much real memory there is and the amount of memory to be occupied by the shared memory pool need to be evaluated carefully. For the formula for computing the shared memory size to be used by the global buffer, see the *HiRDB Version 8 Installation and Design Guide*.
- When the size of the shared memory to be page-fixed is subtracted from the size of the real memory, the result should not equal or exceed half the size that is obtained by subtracting the size of the shared memory to be page-fixed from the swap area size.

Relationship to other facilities

If `fixed` is specified, the shared memory used by a dynamically modified global buffer is also fixed in the real memory. Therefore, carefully consider the real memory size before adding or modifying a global buffer.

Notes

If `fixed` is specified in the AIX 5L version, see the notes for the `pd_shmpool_attribute` operand.

3.3.11 Operands related to statistical information

25) pd_stj_file_size = *maximum-statistics-log-file-size*

~ <unsigned integer>((64-1000000)) (KB)

Specifies in kilobytes the maximum size of a statistics log file.

Two statistics log files are provided. When the amount of stored statistics log information reaches the size specified here, the statistics log files are swapped.

Specification guidelines

- For details about how to determine the value to be specified for this operand, see *D.1 Formulas for determining size of statistics log file (pd_stj_file_size)*.
- Specify this operand such that the following relationship with the

pd_stj_buff_size operand is maintained:

pd_stj_file_size ≥ pd_stj_buff_size × 2

26) pd_stj_buff_size = ***statistics-log-buffer-size***

~ <unsigned integer>((32-512)) (KB)

Specifies the statistics log buffer size.

Specification guidelines

The default value of 32 is appropriate when the following types of statistical information are not to be output:

- SQL object execution information
- Statistics about SQL object transmission

If these types of statistical information are to be output, specify the value obtained by adding 32 to the result obtained from the following formula (however, if the result exceeds 512, specify 512):

$(a \div 1024) \times (0.03 \div b)$

a: Statistics log output volume (bytes)

For details, see *D.1 Formulas for determining size of statistics log file (pd_stj_file_size).*

b: Statistical information output time (seconds)

3.3.12 Operands related to RPC trace information

27) pd_rpc_trace = Y | N

Specifies whether or not RPC trace information is to be collected. HiRDB maintenance information is output in the RPC trace information.

Normally, this operand should be omitted.

Y: Collect RPC trace information.

N: Do not collect RPC trace information.

Note

Specifying Y for this operand degrades communication performance.

28) pd_rpc_trace_name = ***"name-for-RPC-trace-collection-files"***

~ <pathname of up to 254 characters>

Specifies an absolute pathname for the filename for the RPC trace files. Three RPC trace files are created, with 1, 2, and *l* suffixed to the specified file name.

Note

Files with a maximum size of `pd_rpc_trace_size value × 2` are created under the directory specified by this operand. Attention should be paid to the file capacity.

29) pd_rpc_trace_size = *RPC-trace-collection-file-size*

～ <unsigned integer>((1024-2147483648)) (Bytes)

Specifies the size of the RPC trace files in bytes.

The value specified by this operand determines the size of both RPC trace file 1 and RPC trace file 2.

Specification guidelines

An RPC trace collects in a single file in time order the information on the communications among all processes. If the volume of this information exceeds the specified value, trace data output is switched to the other file, which means that older trace information will be overwritten. When this happens, the amount of trace information that is available may be inadequate, making troubleshooting difficult. For this reason, at least 1,000,000 should be specified for this operand.

Note

The value specified by this operand does not apply to RPC trace file *l*, because *l*'s size is fixed at 0 bytes.

3.3.13 Operands related to troubleshooting information

30) pd_cancel_dump = put | noput

This operand is designed to reduce the amount of troubleshooting information to be output.

Specifies whether or not troubleshooting information is to be collected in the following cases:

- When a UAP does not terminate within the monitoring time specified by the `PDCWAITTIME` operands in the client environment definition
- When a UAP being executed is canceled by the `pdcancel` command

For details about the troubleshooting information to be collected, see the *HiRDB Version 8 System Operation Guide.*

`put`:

Troubleshooting information is collected. Because the troubleshooting information is output to files under `$PDDIR/spool`, a space shortage may occur in the file system.

Note that the troubleshooting information that has been collected is automatically deleted by HiRDB at the following timings.

- Every 24 hours while HiRDB is running (the deletion interval can be changed using the `pd_spool_cleanup_interval` operand).
- When HiRDB is started (whether to delete the troubleshooting information can be changed using the `pd_spool_cleanup` operand).

If the HiRDB administrator is to delete the troubleshooting information, the `pdcspool` command must be executed.

`noput`:

Do not collect troubleshooting information. Because no troubleshooting information will be collected, the load on the file system is reduced. Specify this option if UAP cancellation occurs frequently during normal operation and there is no need to investigate the causes.

For details about the error information that is displayed in the event of abnormal termination, see Table *2-1 Error information that is displayed in the event of abnormal termination* in the section on the `pd_dump_suppress_watch_time` operand.

Notes

If a UAP is canceled by the `pdcancel` command with the `-d` option specified, troubleshooting information is collected regardless of the option specified for this operand.

31) pd_dump_suppress_watch_time = *troubleshooting-information-output-suppression-time*

～ <unsigned integer>((0-3600)) (Seconds)

This operand is designed to reduce the amount of troubleshooting information to be output.

This operand specifies the amount of time (in seconds) during which to suppress outputting again the troubleshooting information (files under `$PDDIR/spool`) that is output when any of the following situations occurs.

- The time specified in `PDCWAITTIME` is exceeded.
- The UAP being executed is cancelled by the `pdcancel` command (except when the `-d` option is specified).
- A process is abnormally terminated.

Once troubleshooting information is output, no troubleshooting information is output again until the time specified by this operand has elapsed. For example, if `60` is specified for this operand, no troubleshooting information is output again until 60 seconds have passed because troubleshooting information was previously output.

Note that if `0` is specified for this operand, outputting of troubleshooting information is not suppressed.

Advantage

If there are multiple HiRDB server processes, they may be abnormally terminated continuously because of timeout, for example. When abnormal terminations of server processes occur continuously, troubleshooting information, such as core and simple dump, is repeatedly collected, thus causing a space shortage in the disk in which the HiRDB directory is located. If such a shortage occurs, HiRDB may be abnormally terminated. Therefore, specify this operand to make sure that no disk space shortage occurs.

For details about the error information that is displayed in the event of abnormal termination, see Table *2-1 Error information that is displayed in the event of abnormal termination* in the section on the `pd_dump_suppress_watch_time` operand.

Notes

If the `-d` option is specified for the `pdcancel` command, if abnormal termination is caused by an internal conflict, or if a signal is received from outside, troubleshooting information is collected regardless of the value specified for this operand.

32) pd_spool_cleanup_interval = *troubleshooting-information-deletion-interval*

~ <unsigned integer>((0-744)) (Times)

This operand is used for deleting the troubleshooting information and temporary work files that have been output. If these items are not deleted, they may cause a space shortage in the disk in which the HiRDB directory is located. If such a shortage occurs, HiRDB may be abnormally terminated. Therefore, HiRDB regularly deletes the following files:

- Troubleshooting information file (files in `$PDDIR/spool`)
- Temporary work files (files in `$PDDIR/tmp`)

This operand specifies the deletion interval (hours). For example, if `48` is specified for this operand, these files are deleted every 48 hours. Normally, (if this operand is omitted) files are deleted every 24 hours.

Note that time counting begins when HiRDB is normally started. When HiRDB is normally terminated, time counting also stops. Then, the count returns to 0 during the next normal startup.

Specify the files to be deleted using the pd_spool_cleanup_interval_level operand explained as follows.

Operand rule

If 0 is specified, files are not deleted.

Specification guidelines

If 24, 48, 72, and so on are specified for this operand, files are deleted at the predetermined time. Specify the time so that files are deleted during the time period that does not overload the system.

Notes

- Even while HiRDB is stopped because of planned termination, forced termination, or abnormal termination, time counting continues. However, if the deletion time arrives while HiRDB is stopped, files are not deleted. Files are not deleted until the next deletion time. To restart HiRDB after deleting the files, execute the pdcspool command.
- If the TMPDIR environment variable is specified, the temporary work files used by a command or utility are not output to $PDDIR/tmp, but to the directory specified by the TMPDIR environment variable. The temporary work files that are output to the directory specified by the TMPDIR environment variable are not subject to regular deletion. Therefore, use the rm command of the OS, for example, to delete them.

Remarks

The difference between the pd_spool_cleanup_interval and pd_spool_cleanup operands is as follows:

- The pd_spool_cleanup_interval operand is related to regular deletion of troubleshooting information.
- The pd_spool_cleanup operand is related to the deletion of troubleshooting information during HiRDB startup.

Therefore, if you plan to run HiRDB continuously for 24 hours, consider specifying the pd_spool_cleanup_interval operand. If you plan to terminate HiRDB every day, consider specifying the pd_spool_cleanup operand.

33) pd_spool_cleanup_interval_level = *number-of-days* [, *deletion-type*]

This operand is used for deleting the troubleshooting information and temporary work files that have been output, and specifies the condition for regularly deleting the troubleshooting information and temporary work files.

number-of-days: **~ <unsigned integer>((1-24855)) (days)**

Troubleshooting information files that are older than the number of days specified here are deleted. For example, if 3 is specified, all troubleshooting information files, except for those created within the last 3 days (or 3 days × 24 hours = 72 hours), are deleted.

deletion-type: **<character string>**

Specifies the type of troubleshooting information file to be deleted.

all: All files are to be deleted.

dump: Only the files internally acquired by HiRDB are to be deleted.

The following are the types of troubleshooting information files that are deleted.

Troubleshooting information file type	Directory name	all	dump	Remarks
Deadlock and timeout information	pdlckinf	Y	N	Output when an error occurs during locking.
Access path information	pdsqldump	Y	N	Output when the access path display utility is used.
Save core file, etc.	save	Y	Y	Output when a process is abnormally terminated.
Shared memory dump file	pdshmdump	Y	Y	Output when a process or unit is abnormally terminated.
Simple dump files	pdsysdump	Y	Y	None
	pdsdsdump	Y	Y	Nonexistent in a HiRDB/Parallel Server
	pdfesdump pddicdump pdbesdump	Y	Y	Nonexistent in a HiRDB/Single Server
System log file status information file	pdjnlinf	Y	N	Files under /pdjnlinf/errinf are not deleted.
Transaction information file	pdtrninf	Y	N	Output when Real Time SAN Replication is used.

Y: File is deleted.

N: File is not deleted.

Note

Directory names under $PDDIR/spool are shown.

All temporary work files, except for those listed as follows, are deleted regardless of the deletion type specification. Parentheses indicate directory names under $PDDIR/tmp.

- Current working directory (home) of the process in which HiRDB is to start
- Shared memory information file (pdommenv)
- Differential information files of the pdbufls command (files with names that begin with CMb)

Condition

A value other than 0 must be specified for the pd_spool_cleanup_interval operand.

Specification guidelines

Specify a value that is longer than the execution time of commands (including utilities). For example, if the execution of the pdcopy command, which collects backup data, requires 24 hours (1 day), specify at least 2 for the number of days. If you do not specify a value that is longer than the execution time of the command, the temporary work files being used by the command are deleted, and thus the command may not run correctly.

Operand rule

If you specify a deletion type, you must also specify a number of days.

Remarks

The difference between the pd_spool_cleanup_interval_level and pd_spool_cleanup_level operands is as follows:

- The pd_spool_cleanup_interval_level operand is related to regular deletion of troubleshooting information.
- The pd_spool_cleanup_level operand is related to the deletion of troubleshooting information during HiRDB startup.

Therefore, if you plan to run HiRDB continuously for 24 hours, consider specifying the pd_spool_cleanup_interval_level operand. If you plan to terminate HiRDB every day, consider specifying the pd_spool_cleanup_level operand.

34) pd_spool_cleanup = normal | force | no

This operand is used for deleting the troubleshooting information that has been output.

Specifies whether or not troubleshooting information files (files under $PDDIR/spool) that were output previously by HiRDB are to be deleted when HiRDB is started. This operand is related to the pd_spool_cleanup_level operand described below.

`normal`:

Delete the files when HiRDB is started normally or is restarted following a planned termination.

`force`:

Delete the files whenever HiRDB is started, regardless of the HiRDB activation mode.

`no`:

Do not delete the files.

Specification guidelines

If troubleshooting information files take up too much disk space, specify `normal` or `force`.

Note

Troubleshooting information that is output by a command or utility executed by a user other than the HiRDB administrator may not be deleted. In this case, a user who is authorized to delete troubleshooting information files will have to use the OS's `rm` command to delete the files.

Remarks

The difference between the `pd_spool_cleanup_interval` and `pd_spool_cleanup` operands is as follows:

- The `pd_spool_cleanup_interval` operand is related to regular deletion of troubleshooting information.
- The `pd_spool_cleanup` operand is related to the deletion of troubleshooting information during HiRDB startup.

Therefore, if you plan to run HiRDB continuously for 24 hours, consider specifying the `pd_spool_cleanup_interval` operand. If you plan to terminate HiRDB every day, consider specifying the `pd_spool_cleanup` operand.

35) pd_spool_cleanup_level = *number-of-days* [, *deletion-type*]

This operand is used for deleting the troubleshooting information that has been output, and specifies the condition for deleting the troubleshooting information files during HiRDB startup.

number-of-days: **~ <unsigned integer>((0-24855)) (days)**

Specifies a number of days when troubleshooting information that is older than the specified number of days is to be deleted. For example, if `3` is specified, all troubleshooting information will be deleted except for the information that is fewer than 3 days old (3 days × 24 hours = 72 hours).

If `0` is specified, all troubleshooting information files are deleted.

deletion-type: **<character string>**

Specifies the type of troubleshooting information to be deleted.

`all`: Delete all file types.

`dump`: Delete only files collected internally by HiRDB.

The following are the types of troubleshooting information files that are deleted:

Troubleshooting information file type	Directory name	all	dump	Remarks
Deadlock and timeout information	`pdlckinf`	Y	N	Output when an error occurs during locking.
Access path information	`pdsqldump`	Y	N	Output when the access path display utility is used.
Save core file, etc.	`save`	Y	Y	Output when a process is abnormally terminated.
Shared memory dump file	`pdshmdump`	Y	Y	Output when a process or unit is abnormally terminated.
Simple dump files	`pdsysdump`	Y	Y	None
	`pdsdsdump`	Y	Y	Nonexistent in a HiRDB/Parallel Server

	`pdfesdump`
	`pddicdump`
	`pdbesdump`
		Y	Y	Nonexistent in a HiRDB/Single Server

System log file status information file	`pdjnlinf`	Y	N	Files under `/pdjnlinf/errinf` are not deleted.
Transaction information file	`pdtrninf`	Y	N	Output when Real Time SAN Replication is used.

Y: File is deleted.

N: File is not deleted.

Note

Directory names under `$PDDIR/spool` are shown.

Condition

normal or force (default value) must be specified for the pd_spool_cleanup operand.

Operand rule

A number of days and a deletion type must both be specified.

Remarks

The difference between the pd_spool_cleanup_interval_level and pd_spool_cleanup_level operands is as follows:

- The pd_spool_cleanup_interval_level operand is related to regular deletion of troubleshooting information.
- The pd_spool_cleanup_level operand is related to the deletion of troubleshooting information during HiRDB startup.

Therefore, if you plan to run HiRDB continuously for 24 hours, consider specifying the pd_spool_cleanup_interval_level operand. If you plan to terminate HiRDB every day, consider specifying the pd_spool_cleanup_level operand.

36) pd_module_trace_max = *maximum-number-of-module-traces-that-can-be-stored*

～ <unsigned integer>((126-16383))

A HiRDB process records the history of the executed functions and macros inside the process private memory. This history is called a *module trace*. This operand specifies the number of module trace records. The content of this history is loaded into the core file and is output when a process error occurs.

Specification guidelines

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

Process private memory of the following size is allocated to each process:

In the 32-bit mode: 64 + 48 × pd_module_trace_max *operand value* (bytes)

In the 64-bit mode: 64 + 64 × pd_module_trace_max *operand value* (bytes)

37) pd_module_trace_timer_level = 0 | 10 | 20

Specifies how to acquire the time to be output in module traces. The following table explains the meaning of the value specified for this operand.

Specified value	Time acquisition method
0	Time is output in seconds at every module trace output location.
10	Time is output in microseconds only at performance-critical module trace output locations, such as those before and after input/output processing, and time is output in seconds at other locations.
20	Time is output in microseconds at every module trace output location.

Specification guidelines

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

If you specify a value other than 0 for this operand, a function for acquiring time in microseconds is issued, and as a result, system performance may decline.

3.3.14 Operands related to global buffers

38) pd_dbbuff_wait_interval = ***global-buffer-occupation-state-check-interval***

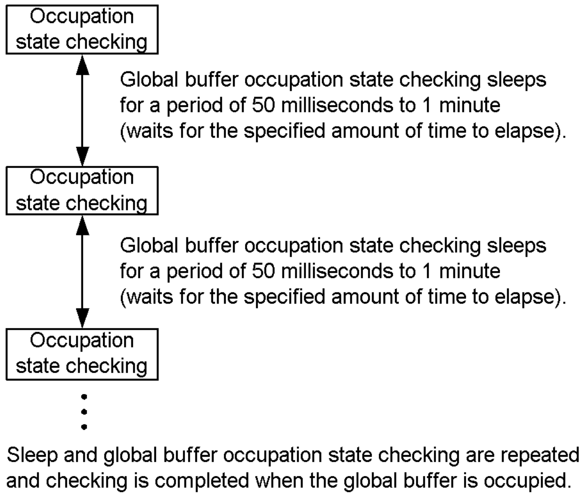
~ **<unsigned integer>((0-2147483647))(milliseconds)**

This operand applies only to a HiRDB/Parallel Server.

Specifies the interval at which the global buffer occupation state is to be checked. Specifying this operand changes the method of checking the global buffer occupation state.

When this operand is not specified

The following processing occurs:


Occupation state checking
Occupation state checking
Occupation state checking

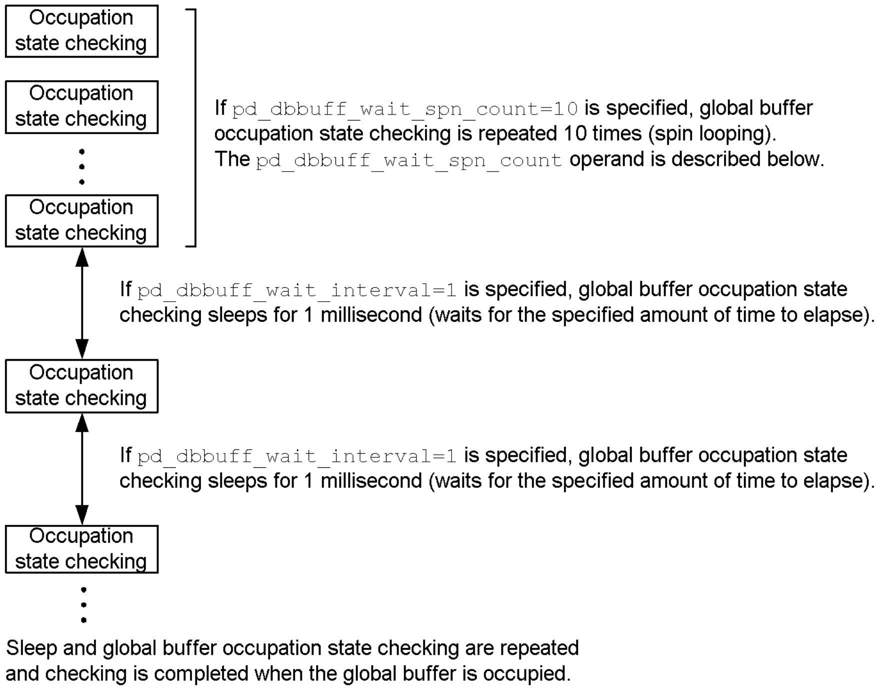
Global buffer occupation state checking sleeps for a period of 50 milliseconds to 1 minute (waits for the specified amount of time to elapse).

Global buffer occupation state checking sleeps for a period of 50 milliseconds to 1 minute (waits for the specified amount of time to elapse).

Sleep and global buffer occupation state checking are repeated and checking is completed when the global buffer is occupied.

When this operand is specified

The following processing occurs:


Occupation state checking
Occupation state checking
Occupation state checking
Occupation state checking
Occupation state checking

If pd_dbbuff_wait_spn_count=10 is specified, global buffer occupation state checking is repeated 10 times (spin looping). The pd_dbbuff_wait_spn_count operand is described below.

If pd_dbbuff_wait_interval=1 is specified, global buffer occupation state checking sleeps for 1 millisecond (waits for the specified amount of time to elapse).

If pd_dbbuff_wait_interval=1 is specified, global buffer occupation state checking sleeps for 1 millisecond (waits for the specified amount of time to elapse).

Sleep and global buffer occupation state checking are repeated and checking is completed when the global buffer is occupied.

Specification guidelines

Specify this operand when all of the conditions listed below are satisfied. Performance may improve. Typically when this operand is used, a value of 1 is specified.

- Global buffer lock-release wait has occurred. (You can check for this based on `WAITL` in the execution result of the `pdbufls` command.)
- You want to improve performance even if the CPU usage rate must be increased.

If the CPU usage has become too high because 1 was specified in this operand, increase the value. If there is unused capacity in the CPU usage rate when 1 is specified in this operand, increase the `pd_dbbuff_wait_spn_count` operand value. Performance may improve.

39) pd_dbbuff_wait_spn_count = ***maximum-spin-loop-count-for-global-buffer-occupation-state-checking***

~ <unsigned integer>((0-2147483646))

This operand is applicable to a HiRDB/Parallel Server.

Specifies the maximum spin loop count in an interval loop that can occur during global buffer occupation state checking. For details, see the description of the `pd_dbbuff_wait_interval` operand.

Specification guidelines

Normally, there is no need to specify this operand. Specify this operand when you specify 1 in the `pd_dbbuff_wait_interval` operand.

3.3.15 Operands related to unit status files

40) pd_syssts_file_name_1 = "*logical-file-name*", "*file-a-status-file-name*", "*file-b-status-file-name*"

: . . .

pd_syssts_file_name_7 = "*logical-file-name*", "*file-a-status-file-name*", "*file-b-status-file-name*"

These operands define status files for the unit.

Although the `pd_syssts_file_name_2` to `7` operands can be omitted, the `pd_syssts_file_name_1` operand cannot be omitted.

"*logical-file-name*" ~ <identifier>((1-8 characters))

Specifies the logical file name of a status file for the unit.

When a command that manipulates the status file is to be executed, the logical file name defined here must be specified.

"*file-a-status-file-name*" ~ <pathname>((maximum of 167 characters))

Specifies the name of the File A status file as an absolute pathname.

"*file-b-status-file-name*" ~ <pathname>((maximum of 167 characters))

Specifies the name of the File B status file as an absolute pathname.

Specification guidelines

- The files specified as File A and File B should be status files created with the `pdstsinit` command. If a file that has not been created with the `pdstsinit` command is specified, a virtual status file is created.
- If an error occurs in a status file, HiRDB swaps the status files. If no spare file is available for swapping, HiRDB (or a unit for a HiRDB/ Parallel Server) terminates abnormally. For this reason, defining a large number of status files improves system reliability, but at the expense of increasing the amount of disk space that is required.
- The status files specified for File A and File B must have the same record length and capacity.

Operand rules

- Up to seven instances of this operand may be specified.
- A status file has a dual structure consisting of File A and File B; both must be specified.
- Environment variables cannot be used for the absolute pathnames of the File A and File B file names.
- The same names cannot be specified for the logical file name, the Status File A file name, and the Status File B file name.

Notes

- When HiRDB is started normally, the primary file (the file that was the primary file when HiRDB was terminated) is inherited. However, if there is no current file that can be inherited, for example because all status files have been initialized, the first status file that was specified among those specified by the `pd_syssts_file_name_1` to `7` operands becomes the current file. The remaining files become spare files if they can be opened. Any files that cannot be opened become reserved files.
- When HiRDB is restarted, the primary file (the file that was the primary file when HiRDB was terminated) is inherited.

Using a virtual status file

Specifying a virtual status file makes it possible to add a new status file while

HiRDB is running.

For example, if the number of spare files is reduced because of errors in status files, virtual status files can be converted into spare files. The procedure for converting virtual status files into spare files follows.

Procedure

1. Use the `pdstsinit` command to create a status file in a HiRDB file system area for system files.
2. Use the `pdstsopen` command to open the status file.

These operations can be performed while HiRDB is running; there is no need to stop HiRDB.

Advantages and disadvantages

Defining a virtual status file reduces the amount of space required in the HiRDB file system area. However, a virtual status file cannot be added as a spare file if the HiRDB file system area for system files does not have sufficient space (sufficient space for adding the file), thus reducing system reliability.

When virtual status files are not defined, the amount of space required in the HiRDB file system area is increased. However, because a swap destination is guaranteed when a file error occurs, system reliability is increased.

Notes

When virtual status files are defined, HiRDB determines that a status file error has occurred when the unit is started. For this reason, HiRDB cannot start if `stop` (default value) is specified for the `pd_syssts_initial_error` operand. When virtual status files are defined, specify `continue` or `excontinue` for the `pd_syssts_initial_error` operand. It is also necessary before starting HiRDB to specify the current file in the `pd_syssts_last_active_file` operand.

Relationship to other operands

This operand is related to the `pd_syssts_subfile_name_1` to `7` operand.

41) pd_syssts_subfile_name_1 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"

:

pd_syssts_subfile_name_7 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"

Defines the unit status files for log application processing, which are used at the log application site when Real Time SAN Replication based on the log-only synchronous method is used.

"*logical-file-name*": ～ <identifier> ((1-8 characters)

Specifies the logical file name of a unit status file for log application processing. This must be a logical file name specified in `pd_syssts_file_name_1` through `pd_syssts_file_name_7`. A logical file name defined here is used for executing commands that manipulate the status files for log application processing.

"*primary-status-file-name-for-log-application-processing*": ～ <path name> ((up to 167 characters)

Specifies the absolute path name of a primary status file for log application processing.

"*secondary-status-file-name-for-log-application-processing*": ～ <path name> ((up to 167 characters)

Specifies the absolute path name of a secondary status file for log application processing.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.

Specification guidelines

- For the primary and secondary status file names for log application processing, specify the names of status files for log application processing that were created in the preparations for log application. If any other status file for log application processing is specified, the specified file becomes a *virtual status file for log application processing*.
- If an error occurs in a status file for log application processing, HiRDB swaps status files for log application processing. If no spare file is available for swapping, HiRDB (or the unit in the case of a HiRDB/ Parallel Server) terminates abnormally. For this reason, defining a large number of status files improves system reliability, but at the expense of increasing the amount of disk space that is required.
- The specified primary and secondary status files for log application processing must have the same record length and capacity.

Operand rules

- You can specify a maximum of 7 instances of this operand.
- This operand is ignored at the transaction execution site.
- If you performed the preparations for log application during HiRDB startup, create status files for log application processing that correspond to all status files for transaction processing that can be opened and that were specified in `pd_syssts_file_name_1` through `pd_syssts_file_name_7`.
- The system uses dual status files for log application processing. Make sure that you specify both files.
- Environment variables cannot be used for the absolute path names of the primary and secondary status files for log application processing.
- Each logical file name, primary status file name for log application processing, and secondary status file name for log application processing must be unique.
- The names of the primary and secondary status files for log application processing must be different from the primary and secondary status file names specified in the `pd_syssts_file_name_1` through `pd_syssts_file_name_7` operands.

Notes

When HiRDB is started normally or is restarted, the current file (the file that was the current file when HiRDB last terminated) is inherited. However, if there is no current file that can be inherited, such as when all status files for log application processing have been initialized, the log application site can no longer be started. If this occurs, perform the preparations for log application.

How to use a virtual status file for log application processing

See the description of the `pd_syssts_file_name_1` through `pd_syssts_file_name_7` operands.

Relationship to other operands

This operand is related to the `pd_syssts_file_name_1` through `pd_syssts_file_name_7` operands.

3.3.16 Operands related to unit status files (when an error occurs)

For details about the measures to be taken when an error occurs in a status file, see the *HiRDB Version 8 System Operation Guide*.

42) pd_syssts_initial_error = <u>stop</u> | continue | excontinue

When HiRDB (or a unit) starts, HiRDB performs a process of identifying the current unit status file. This operand specifies the action that HiRDB takes when any of the following errors is detected during this identification process.

- No real unit status file is found.
- An error is detected in the unit status file.

Note that the current file identification process is applied to the unit status files specified by the pd_syssts_file_name_1 to 7 operands.

If Real Time SAN Replication based on the log-only synchronous method is used, the unit status files for log application processing specified in the pd_syssts_subfile_name1 through pd_syssts_subfile_name7 operands are also used at the log application site. For the log application site, replace the pd_syssts_last_active_file and pd_syssts_last_active_side operands in the above description with the pd_syssts_last_active_subfile and pd_syssts_last_active_side_sub operands, respectively.

stop:

When an error is detected in the unit status file during the current file identification process, the startup of HiRDB (or the unit) is stopped. In this case, first take a corrective action for the status file in which the error was detected, and then start HiRDB.

continue or excontinue:

Even when an error is detected in the unit status file during the current file identification process, the startup of HiRDB (or the unit) is continued if the current file is normal. However, the startup may be stopped depending on the value specified for the pd_syssts_singleoperation operand (whether operation should continue with a single status file). The following table shows the relationship to the pd_syssts_singleoperation operand.

- **Relationship to the pd_syssts_singleoperation operand**

pd_syssts_singleoperation operand value	Processing by HiRDB	HiRDB administrator's action
continue	When an error is detected in the unit status file, HiRDB cannot identify the current file, and thus the startup of HiRDB (or the unit) is stopped.	The HiRDB administrator identifies the current file and specifies the pd_syssts_last_active_file and pd_syssts_last_active_side operands. Afterwards, start HiRDB (or the unit).

pd_syssts_singleoperation operand value	Processing by HiRDB	HiRDB administrator's action
stop (default value)	When an error is detected in the unit status file, HiRDB identifies the current file, and the startup of HiRDB (or the unit) is continued. However, if File A or B satisfies any of the conditions listed in the following table (cases in which HiRDB cannot identify the current file), the startup of HiRDB is stopped.	If HiRDB cannot identify the current file, the HiRDB administrator identifies the current file and specifies the pd_syssts_last_active_file and pd_syssts_last_active_side operands. Afterwards, start HiRDB (or the unit).

- **When HiRDB cannot identify the current file**

pd_syssts_initial_error operand value	File A status	File B status
continue	Error shutdown	Error shutdown
	Error shutdown	Open (initial state)
	Error shutdown	No real file
	Open (initial state)	Error shutdown
	Open (initial state)	No real file
	No real file	Open (initial state)
	No real file	Error shutdown
	No real file	No real file
excontinue	Error shutdown	Error shutdown
	Error shutdown	No real file
	No real file	Error shutdown
	No real file	No real file

Therefore, to minimize actions by the HiRDB administrator (to reduce the number of cases in which both the pd_syssts_last_active_file and pd_syssts_last_active_side operands must be specified), specify the following:

- pd_syssts_initial_error = excontinue
- pd_syssts_singleoperation = stop

Specification guidelines

The following table shows the specification guidelines, along with their advantages and disadvantages.

Item	pd_syssts_initial_error operand value	
	stop	continue or excontinue
Processing by HiRDB during HiRDB or unit startup	When an error is detected in a unit status file, the startup of HiRDB (or the unit) is stopped.	Even when an error is detected in a unit status file, the startup of HiRDB (or the unit) is continued if the current file is normal.
Specification guideline	To improve system reliability, specify `stop`.	To simplify the error-handling actions during HiRDB startup, specify `continue` or `excontinue`.
Advantage	Guarantees that all unit status files are normal when HiRDB starts. Therefore, if an error occurs in the current file after HiRDB has started, it can be swapped to a spare file.	Even when an error is detected in a unit status file during HiRDB startup, HiRDB can start with the remaining normal files only. Therefore, HiRDB stop time can be shortened. In this case, because the number of spare files has become small, it is necessary to immediately repair the status files containing errors.
Disadvantage	Possibility increases that an error in a unit status file stops the startup of HiRDB.	Because HiRDB may be running with only a small number of spare files, system reliability is low. Depending on the number of spare files available, it may not be possible to swap unit status files.

Notes

- If both current files are abnormal, the startup of HiRDB (or the unit) is stopped regardless of the value specified for this operand.
- Before starting HiRDB, do not initialize the current file with the `pdstsinit` command. If the current file is initialized, HiRDB cannot be restarted.

Remarks

The following figure shows specification values for this operand, processing by HiRDB, and actions to be taken by the HiRDB administrator.

- Specification values for this operand and processing by HiRDB

`pd_syssts_initial_error` operand value	Status file error	`pd_syssts_single operation` operand value	Can HiRDB identify the current file?	Specification of `pd_syssts_last_active_file`	Does the value of the `pd_syssts_last_active_file` operand match the latest file that can be opened?	Current file error	Specification of the `pd_syssts_last_active_side` operand	Is the file specified for the `pd_syssts_last_active_side` operand usable?	Numbers indicated in the Table *Processing by HiRDB and actions by HiRDB administrator*
`stop` (default value)	None	—	—	—	—	—	—	—	[1]
	Yes	—	—	—	—	—	—	—	[5]
`continue`	None	—	—	—	—	—	—	—	[1]
	Yes	`stop` (default value)	Can be ID'd.	—	—	—	—	—	[2]
			Cannot be ID'd.	None	—	—	—	—	[6]
				Yes	Matches.	None	—	—	[3]
						Yes	None	—	[9]
							Yes	Usable	[4]
								Not usable	[7]
					Does not match.	—	—	—	[8]
		`continue`	Can be ID'd.	—	—	—	—	—	Does not occur.
			Cannot be ID'd.	None	—	—	—	—	[6]
				Yes	Matches.	None	—	—	[3]
						Yes	None	—	[9]
							Yes	Usable	[4]
								Not usable	[7]
					Does not match.	—	—	—	[8]
`excontinue`	None	—	—	—	—	—	—	—	[1]
	Yes	`stop` (default value)	Can be ID'd.	—	—	—	—	—	[2]
			Cannot be ID'd.	None	—	—	—	—	[6]
				Yes	Matches.	None	—	—	[3]
						Yes	None	—	[9]
							Yes	Usable	[4]
								Not usable	[7]
					Does not match.	—	—	—	[8]
		`continue`	Can be ID'd.	—	—	—	—	—	Does not occur.
			Cannot be ID'd.	None	—	—	—	—	[6]
				Yes	Matches.	None	—	—	[3]
						Yes	None	—	[9]
							Yes	Usable	[4]
								Not usable	[7]
					Does not match.	—	—	—	[8]

—: Not applicable. (The condition does not affect the processing by HiRDB.)

- Processing by HiRDB and actions by HiRDB administrator

Corresponding No.	Processing by HiRDB	HiRDB administrator's action
[1]	HiRDB startup processing continues.	None
[2]	HiRDB identifies the latest current file and continues the startup processing.	Make the file that is in the error-shutdown state into a spare file.
[3]	Using the file specified in the `pd_syssts_last_active_file` operand as the current status file, HiRDB startup processing continues.	Make the file that is in the error-shutdown state into a spare file.

Corresponding No.	Processing by HiRDB	HiRDB administrator's action
[4]	Using the files specified in the `pd_syssts_last_active_file` and `pd_syssts_last_active_side` operands as the current status files, HiRDB startup processing continues.	Make the file that is in the error-shutdown state into a spare file.
[5]	Because `stop` is specified for the `pd_syssts_initial_error` operand, HiRDB startup processing is stopped.	See the manual *HiRDB Version 8 Messages*, and take the corrective action indicated by the reason code `0000000010` in message `KFPS01005-E`.
[6]	Because the current file that was being used during the previous operation cannot be identified, HiRDB startup processing is stopped.	See the manual *HiRDB Version 8 Messages*, and take the corrective action indicated by the reason code `0000000016` in message `KFPS01005-E`.
[7]	Because the normal current file identified by HiRDB does not match the file specified in the `pd_syssts_last_active_side` operand, HiRDB startup processing is stopped.	See the manual *HiRDB Version 8 Messages*, and take the corrective action indicated by the reason code `0000000017` in message `KFPS01005-E`.
[8]	Because the current file name identified by HiRDB does not match the file name specified in the `pd_syssts_last_active_file` operand, HiRDB startup processing is stopped.	See the manual *HiRDB Version 8 Messages*, and take the corrective action indicated by the reason code `0000000015` in message `KFPS01005-E`.
[9]	Because the normal current file that was being used during the previous operation cannot be identified, HiRDB startup processing is stopped.	See the manual *HiRDB Version 8 Messages*, and take the corrective action indicated by the reason code `0000000018` in message `KFPS01005-E`.

43) pd_syssts_singleoperation = stop | continue

Specifies whether or not processing may continue in the single-operation mode for unit status files.

The status file single-operation mode means that processing is to continue using only the normal file (single file) when an error occurs in the status file and a spare file is not available. For details about the status file single-operation mode, see the *HiRDB Version 8 Installation and Design Guide*.

If an error occurs in one of the current files and a spare file is available, the status file is swapped and processing continues regardless of the value specified for this operand (operation in the single-operation mode does not occur).

If Real Time SAN Replication based on the log-only synchronous method is used, the unit status files for log application processing are also used.

`stop`:

Do not permit operation in the single-operation mode. If operation in the single-operation mode is necessary, HiRDB (or a unit for a HiRDB/Parallel Server) is abnormally terminated. If HiRDB is abnormally terminated, allocate spare files and then restart HiRDB.

`continue`:

Enable operation in the single-operation mode. When the single-operation mode goes into effect, the message `KFPS01044-I` is output. If an error occurs in the normal file during operation in the single-operation mode, or if HiRDB (or a unit for a HiRDB/Parallel Server) is abnormally terminated while the status file is being updated, HiRDB cannot be restarted. Therefore, when the single-operation mode goes into effect, immediately allocate spare files.

Specification guidelines

- Hitachi recommends that you specify `stop` to increase system reliability. Hitachi also recommends that you increase the number of spare files to guard against errors in the current files.
- The following table shows the specification guidelines, along with their advantages and disadvantages.

Item	pd_syssts_singleoperation operand value	
	stop	continue
Specification guideline	To improve system reliability, specify `stop`.	Specify `continue` if it is important not to stop HiRDB.
Advantage	When an error occurs in one of the current files and a spare file is not available, operation in the single-operation mode does not occur, and HiRDB is abnormally terminated. Consequently, the possibility of losing the content of the current files is reduced.	Even when an error occurs in one of the current files and a spare file is not available, processing can be continued. Therefore, the possibility that an error in the status file stops HiRDB is reduced.
Disadvantage	Possibility that an error in the status file stops HiRDB increases. However, increasing the number of spare files can reduce this possibility.	If an error occurs in the normal status file during operation in the single-operation mode, or if HiRDB is abnormally terminated during updating of the status file, the content of the current file is lost and HiRDB cannot be restarted.

Relationship to other operands

The combination of the values specified for the pd_syssts_singleoperation and pd_syssts_initial_error operands determines the processing by HiRDB when an error occurs in the status file. Therefore, the values to be specified for these operands should be determined together.

44) pd_syssts_last_active_file = "*logical-file-name*"

~ <identifier>((1-8 characters))

Specifies the name of the logical file name of the status file to be used as the current file at the time of HiRDB (or a unit for a HiRDB/Parallel Server) startup.

HiRDB compares the file specified in this operand with the file selected by HiRDB to be the current file. If they match, HiRDB is started; otherwise, HiRDB is not started.

Conditions

The following conditions must be satisfied:

- continue or excontinue must be specified for the pd_syssts_initial_error operand.
- It cannot be determined if the current file selected by the HiRDB system was the most recent current file during the previous session.

Specification guidelines

1. To start HiRDB immediately after initializing all status files:

   From among the operable logical files specified in the pd_syssts_file_name_1-7 operands, specify the one that has the smallest number. Forced startup will be used in this case, regardless of the previous termination mode.

2. When both of the current status files are normal:

   Specify the name of the current file*.

   If HiRDB cannot be started even though the name of the current file is specified, the current file may have been initialized. In this case, first initialize all status files, then use the method in 1 above to start HiRDB (forced startup will be used, regardless of the previous termination mode).

3. When one of the current status files has an error:

   Use the method in 2 above, with the following operands specified:

   - pd_syssts_singleoperation = continue or excontinue

- pd_syssts_last_active_side

4. When both of the current status files have errors:

   Initialize all status files, then execute the method in 1 above (forced startup will be used, regardless of the previous termination mode).

5. When a virtual status file is specified

   Specify the name of the current file*.

* The names of the current files (that were active at the end of the previous operation) can be determined from the following messages:

- KFPS01001-I
- KFPS01010-E
- KFPS01011-I
- KFPS01063-I

Of the status files displayed by these messages, the one that is reported in the message that was output most recently is the current file.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the unit status files for transaction processing are used at the transaction execution site. At the log application site, this operand is ignored, if specified.

45) pd_syssts_last_active_side = A | B

Specify this operand if you want to start HiRDB (or a unit for a HiRDB/Parallel Server) when one of the current files is in an error state. Specify the normal status file for this operand. HiRDB compares the file specified in this operand with the file selected by HiRDB. If they match, HiRDB copies the contents of the normal status file to secondary File A and File B. Afterwards, the secondary file is switched to the current file and HiRDB is started. If the files do not match, HiRDB is not started.

Conditions

The following operands must be specified:

- pd_syssts_initial_error = continue or excontinue
- pd_syssts_last_active_file

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the status files for transaction processing are used at the transaction execution site. At the log application site, this operand is ignored, if specified.

46) pd_syssts_last_active_subfile = "*logical-file-name*"

~ <identifier> ((1-8 characters))

When Real Time SAN Replication based on the log-only synchronous method is used, this operand specifies the logical file name of the status file for log application processing that is to be used as the current file when HiRDB (or the unit in the case of a HiRDB/Parallel Server) starts. HiRDB compares the file specified in this operand with the current file selected by HiRDB. If the file names match, HiRDB starts; if not, HiRDB does not start.

Conditions

For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*

The following conditions must be satisfied:

- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.
- `continue` or `excontinue` is specified in the `pd_syssts_initial_error` operand.
- It cannot be determined if the current file selected by the HiRDB system was the most recent current file during the previous session.

Specification guidelines

1. When starting HiRDB immediately after initializing all status files for log application processing

   Perform the preparations for log application. Once the preparations for log application have been completed successfully, start HiRDB without specifying this operand.

2. When both of the current files are normal

   Specify the name of the current file.# If HiRDB cannot be started even though the name of the current file is specified, the current file may have been initialized. In such a case, first initialize all status files for log application processing, then use the method in 1 above to start HiRDB. Note that in this case, forced startup will be used, regardless of the previous termination mode.

3. When one of the current files has an error

   Use the method in 2 above and specify the following operands:

- Specify `continue` in the `pd_syssts_singleoperation` operand
- Specify the `pd_syssts_last_active_side_sub` operand

4. When both of the current files have errors

   Perform the preparations for log application. Once the preparations for log application have been completed successfully, start HiRDB without specifying this operand.

5. When a virtual status file for log application processing is specified

   Specify the name of the current file.#

#

The names of the current files (that were active at the end of the previous session) can be determined from the following messages:

- `KFPS01001-I`
- `KFPS01010-E`
- `KFPS01011-I`
- `KFPS01063-I`

Of the status files for log application processing that are displayed in these messages, the one that is reported in the message that was output most recently is the current file.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the unit status files for log application processing are used at the log application site. At the transaction execution site, this operand is ignored, if specified.

47) pd_syssts_last_active_side_sub = A | B

When Real Time SAN Replication based on the log-only synchronous method is used, specify this operand if you wish to start HiRDB (or the unit in the case of a HiRDB/Parallel Server) while one of the current files is in error status. In such a case, you specify the normal file in this operand. HiRDB compares the file specified in this operand with the file selected by HiRDB. If their names match, HiRDB copies the contents of the normal file into the secondary file. HiRDB then designates the secondary file as the current file and starts itself. If the file names do not match, HiRDB does not start.

Conditions

For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*

The following conditions must be satisfied:

- Y must be specified in the pd_rise_use operand, and also syssync must be specified in the pd_rise_pairvolume_combination operand.
- continue or excontinue is specified in the pd_syssts_initial_error operand.
- The pd_syssts_last_active_subfile operand is specified.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the unit status files for log application processing are used at the log application site. At the transaction execution site, this operand is ignored, if specified.

3.3.17 Operands related to security audit facility

For the method of using the security audit facility, see the *HiRDB Version 8 System Operation Guide*.

48) pd_audit = Y | N

Specifies whether to begin collecting an audit trail when HiRDB (or a unit for a HiRDB/Parallel Server) is started.

Y: Begins collecting an audit trail when HiRDB is started.

N: Does not begin collecting an audit trail when HiRDB is started.

Even if N is specified for this operand, you can still collect an audit trail by executing the pdaudbegin command.

Conditions

All of the following conditions must be satisfied. If Y is specified when all of these conditions are not satisfied, HiRDB (or a unit for a HiRDB/Parallel Server) cannot be started.

- A HiRDB file system area has been created for an audit trail file.
- The name of the HiRDB file system for the audit trail file is specified for the pd_aud_file_name operand.

49) pd_aud_file_name = *HiRDB-file-system-area-name-for-audit-trail-file*

～ <pathname>((up to 150 characters))

This operand is required if you use the security audit facility. If you do not specify this operand, you cannot use the security audit facility.

Specify an absolute path name for the name of the HiRDB file system area for an audit trail file.

Notes

- When this operand is specified, HiRDB (or a unit for a HiRDB/Parallel Server) cannot be started if an error occurs during the access to the HiRDB file system area for audit trail files.
- For a system configuration in which multiple units run on the same server machine, use this operand to specify an audit trail file for each unit. If the `pd_aud_file_name` operand in the system common definition specifies the same audit trail file for multiple units on the same server machine, audit trails cannot be correctly collected.

  However, for a unit to which the standby-less system switchover (effects distributed) facility is applied, specify this operand in the system common definition.

50) pd_aud_max_generation_size = *audit-trail-file-maximum-size*

～ <unsigned integer>((1-5240)) (MB)

Specifies the maximum size (MB) of audit trail files.

Specification guidelines

Because HiRDB needs 20 MB for management, determine the value for this operand so that the following condition is satisfied:

pd_aud_max_generation_size-value × *pd_aud_max_generation_num-value* < *size-of-HiRDB-file-system-area-for-audit-trail-files* (value of the `-n` option of the `pdfmkfs` command) - 20 MB

51) pd_aud_max_generation_num = *maximum-audit-trail-file-count*

～ <unsigned integer>((2-200))

Specifies the maximum number of (number of generations of) audit trail files to be created inside the HiRDB file system area for audit trail files.

Specification guidelines

- Hitachi recommends that you not specify the maximum value (200) in case errors occur in all audit trail files. For the method of handling errors in audit trail files, see the *HiRDB Version 8 System Operation Guide*.
- Because HiRDB needs 20 MB for management, determine the value for this operand so that the following condition is satisfied:

pd_aud_max_generation_size-value × *pd_aud_max_generation_num-value* < *size-of-HiRDB-file-system-area-for-audit-trail-files* (value of the `-n` option of the `pdfmkfs` command) - 20 MB

Notes

During the startup of HiRDB (or a unit for a HiRDB/Parallel Server), if there is a file with a generation number that is greater than the value specified for this operand, the specified value becomes invalid. In this case, the largest generation number is assumed as the maximum number of audit trail files to be created inside the HiRDB file system area.

52) pd_aud_async_buff_size = ***size-of-buffer-used-for-asynchronous-output-of-audit-trail-file***

~ <unsigned integer>((0, 4096-6553600)) (Byte)

Specifies the size (bytes) of the buffer to be used for asynchronously outputting audit trail. If `0` is specified, audit trail is synchronously output. The following table describes the advantages and disadvantages of each output method.

pd_aud_async_buff_size value	Audit trail output method	Advantages	Disadvantages
`0`	Synchronous output	Audit trail can be reliably output to an audit trail file.	Because file input/out occurs on the extension of SQL processing, the impact on performance is large.
`4096-6553600`	Asynchronous output	Can reduce the impact on SQL processing performance.	If HiRDB (or unit for a HiRDB/Parallel Server) is abnormally terminated after the audit trail is output to the buffer and before it is output to an audit trail file, the audit trail may be lost.

Operand rule

For this operand, specify an integer multiple of 4096. If a value that is not an integer multiple of 4096 is specified, it is rounded up to an integer multiple of 4096 and set as the value for this operand. For example, if `5000` is specified, `8192` is set for the operand.

53) pd_aud_async_buff_count = ***number-of-buffer-sectors-used-for-asynchronous-output-of-audit-trail-file***

~ <unsigned integer>((1-6500))

Specifies the number of buffer sectors to be used for asynchronously outputting an audit trail.

54) pd_aud_async_buff_retry_intvl =

retry-interval-for-allocation-of-a-buffer-to-be-used-for-asynchronous-output-of-audit-trail-file

~ <unsigned integer>((1-1000)) (milliseconds)

Specifies the retry interval for monitoring for a buffer to be used for asynchronous output of the audit trail so that the audit trail can be acquired when all buffers are in use.

Specification guidelines

Normally, there is no need to specify this operand.

When the security audit facility is used and a UAP requires an extended amount of time to execute, specifying a small value in this operand may reduce the UAP execution time.

3.3.18 Operands related to the system switchover facility

55) pd_ha_acttype = <u>monitor</u> | server

Specifies whether to run the system switchover facility in the monitor mode or server mode. For details on the monitor and server modes, see the *HiRDB Version 8 System Operation Guide*.

`monitor`: The system switchover facility runs in the monitor mode.

`server`: The system switchover facility runs in the server mode.

If any of the following facilities is used, the server mode is used as the default.

- User server hot standby
- Rapid system switchover facility
- Standby-less system switchover (1:1) facility
- Standby-less system switchover (effects distributed) facility

Conditions

To specify this operand, you must specify `use` for the `pd_ha` operand. If you specify `server`, all of the following conditions must be satisfied:

- The cluster software used is the HA monitor, MC/ServiceGuard, or VERITAS Cluster Server.
- Hitachi HA Toolkit Extension is installed.

If the cluster software is the HA monitor, Hitachi HA Toolkit Extension is not required. If you specify `server` without installing these products, HiRDB cannot start.

Notes

- If the cluster software is the HA monitor, match the specification of this operand to the specification for the `acttype` operand of the HA monitor. If they are different, HiRDB may not be able to start, or system switchover may not occur, for example.
- If the cluster software is Sun Cluster, ClusterPerfect, or HACMP, do not specify `server` for this operand.

56) pd_ha_unit = nouse

Specify this operand if the system switchover facility is not to be applied to the unit. Specify this operand when the facility is to be applied to only some of the units. Specifying this operand invalidates the specifications of all of the operands explained in this section.

Relationship to other operands

If the `-k stls` option is specified in the front-end server's `pdstart` operand, a recovery-unnecessary front-end server unit is assumed. The system switchover facility is not applicable to a recovery-unnecessary front-end server unit. For a system that employs the system switchover facility, make sure that you specify this operand in the unit control information definition for the recovery-unnecessary front-end server unit.

57) pd_ha_restart_failure = command-name

～ <pathname>

This operand can be specified when the monitor mode is used. Specification of this operand is invalid in the server mode.

Specify the absolute path name of the command to be executed when a HiRDB restart fails. By specifying a command for switching the cluster software system for this operand or by issuing a command for switching the system inside a user command (shell), you can automate the system switchover operation that is to take place when a HiRDB restart fails.

58) pd_ha_switch_timeout = Y | N

This operand can be specified when the server mode is used. Specification of this operand is invalid in the monitor mode.

This operand specifies whether to switch the system without waiting for internal termination processing of HiRDB when internal termination processing of HiRDB (or a unit for a HiRDB/Parallel Server) during system switchover has exceeded the server failure monitoring time. The server failure monitoring time referred to here is the time specified for the `patrol` operand of the HA monitor or Hitachi HA Toolkit Extension.

For details about the `patrol` operand of Hitachi HA Toolkit Extension, see the manual *Hitachi HA Toolkit*.

`Y`:

Switches the system without waiting for internal termination processing of HiRDB when internal termination processing of HiRDB during system switchover has exceeded the server failure monitoring time. In this case, system switchover is carried out by assuming that HiRDB has slowed down.

If you are using the standby-less system switchover (1:1) facility or the standby-less system switchover (effects distributed) facility, the specification for this operand is invalid during planned system switchover.

`N`:

Does not switch the system until internal termination processing of HiRDB during system switchover is terminated.

Advantage

If internal termination processing of HiRDB during system switchover takes a long time, because, for example, of a disk error, system switchover may be delayed as a result. If you specify `Y` (default value) for this operand, you can switch the system without waiting for internal termination processing of HiRDB, even when it is taking a long time.

Notes

- If you specify `Y` for this operand when the `patrol` operand value is small, planned system switchover may turn into system switchover based on slow-down. This is because internal termination processing of HiRDB during planned system switchover exceeds the time specified by the `patrol` operand.
- You need to be careful if `restart` is specified for the `switchtype` operand of the HA monitor or Hitachi HA Toolkit Extension. If `pd_ha_switch_timeout=Y` (default value) is specified, and if internal termination processing of HiRDB exceeds the server failure monitoring time, HiRDB is not started in the system in which the failure occurred. In this case, the system is immediately switched.

  For details about the `switchtype` operand of the Hitachi HA Toolkit Extension, see the manual *Hitachi HA Toolkit*.

59) pd_ha_server_process_standby = Y | N

Specifies whether to use user server hot standby. For details on user server hot standby, see the *HiRDB Version 8 System Operation Guide*.

`Y`: Uses user server hot standby.

`N`: Does not use user server hot standby.

Omit this operand if you use the standby-less system switchover (1:1) facility or standby-less system switchover (effects distributed) facility.

Condition

HiRDB must be in the server mode if you are to specify Y (default value) for this operand. If the monitor mode is in effect, N is always assumed.

Relationship to other operands

The rapid system switchover facility includes user server hot standby. Therefore, if standbyunit is specified (to use the rapid system switchover facility) for the pd_ha_agent operand, Y is assumed for the pd_ha_server_process_standby operand even if you specify N for it.

60) pd_ha_agent = standbyunit | server | activeunits

Specify this operand when you use the following facilities:

- Rapid system switchover facility
- Standby-less system switchover (1:1) facility
- Standby-less system switchover (effects distributed) facility

For details about these facilities, see the *HiRDB Version 8 System Operation Guide*.

standbyunit: Specify this option when you use the rapid system switchover facility.

server: Specify this option when you use the standby-less system switchover (1:1) facility.

activeunits: Specify this option when you use the standby-less system switchover (effects distributed) facility.

Note that you cannot specify both the standby-less system switchover (1:1) facility and the standby-less system switchover (effects distributed) facility within the same system.

Condition

If you specify this operand, the server mode must be used. If the monitor mode is used, the error message KFPS01896-E is output.

Notes

- A unit that is the target of the rapid system switchover facility cannot inherit IP addresses. Therefore, for a HiRDB/Single Server, specify N for the pd_ha_ipaddr_inherit operand in the system common definition or unit control information definition. If a configuration to inherit IP addresses is used for a HiRDB/Parallel Server, specify N for the pd_ha_ipaddr_inherit operand in the unit control information definition of the unit that is the target of the rapid system switchover facility.

- When you use the rapid system switchover facility, standby-less system switchover (1:1) facility, or standby-less system switchover (effects distributed) facility, global buffers cannot be dynamically changed. For details about how to dynamically change global buffers, see the *HiRDB Version 8 System Operation Guide*.

Relationship to other operands

- The rapid system switchover facility includes user server hot standby. Therefore, if you specify `standbyunit` for this operand, `Y` is assumed for the `pd_ha_server_process_standby` operand.
- The standby-less system switchover (1:1) facility and the standby-less system switchover (effects distributed) facility use the server process of the unit at the switching destination following a system switchover. Consequently, if you specify `server` or `activeunits` for this operand, the specification of the `pd_ha_server_process_standby` operand becomes invalid.
- When you use the rapid system switchover facility, standby-less system switchover (1:1) facility, or standby-less system switchover (effects distributed) facility, `Y` is assumed for the `pd_rdarea_open_attribute_use` operand (RDAREA opening trigger specification).
- When you use the standby-less system switchover (effects distributed) facility, not all units within an HA group need to be active, and therefore the specification of the `pd_start_skip_unit` operand becomes invalid. However, all back-end servers within the HA group must be running at some units. Otherwise, HiRDB cannot start.
- When you specify this operand, check the values specified for the following operands. If the specification values are wrong, the error message `KFPS01896-E` is output.

Operand name	pd_ha_agent operand value		
	standbyunit	server	activeunits
`pd_ha`	`use`	`use`	`use`
`pd_ha_unit`	Omitted	Omitted	Omitted
`pd_ha_acttype`	`server`	`server`	`server`
`pd_ha_ipaddr_inherit` (unit control information definition)	`N`	--	--

Operand name	pd_ha_agent operand value		
	standbyunit	server	activeunits
pd_ha_agent	—	activeunits must not be specified for other units.	server must not be specified for other units.
-c option of the pdstart operand	—	Y	—
-g option of the pdstart operand	—	—	Y
pdhagroup	—	—	Y

Legend:

Y: Operand must be specified.

— : Not applicable

61) pd_ha_max_act_guest_servers = *maximum-number-of-acceptable-guest-BES*

~ <unsigned integer>((0-33))

Specify this operand when you use the standby-less system switchover (effects distributed) facility. This operand specifies the maximum number of guest back-end servers that operate as running systems inside the unit. When the number of guest back-end servers that are operating as running systems inside the unit reaches the value specified for this operand, the accepting state for guest back-end servers is cancelled.

The value of this operand determines the guest area to be allocated by the accepting unit. Adjust the resource requirement for back-end servers by changing the value of this operand.

If the value of this operand is too small, processing for some of the back-end servers may not take place when a system switchover occurs. When the number of back-end servers in an error state inside the HA group exceeds the total number of acceptable guest back-end servers inside the HA group, processing for the excess back-end servers is not accepted by any unit, and consequently, processing for these back-end servers does not occur.

Condition

When you specify this operand, you must also specify the following operands. If they are not specified, a system definition error occurs (message KFPS01896-E is output).

- pdhagroup
- pd_ha_agent = activeunits

Specification guidelines

If this operand is omitted, HiRDB calculates a value for this operand. The upper limit of the value that can be specified is the smaller of the following two values:

- 34 - *number-of-host-back-end-servers-inside-unit*
- *number-of-servers-inside-HA-group* - *number-of-host-BESs-inside-unit*

Even if you specify a value greater than the upper limit, it is corrected to the upper limit (message `KFPS05613-W` is output).

Note that the maximum number of acceptable guest back-end servers calculated by HiRDB when this operand is omitted assumes that all of the following conditions are satisfied:

- All units inside the HA group have the same number of back-end servers.
- A 2-point error has occurred.

Therefore, if all units inside the HA group do not have the same number of back-end servers, or if a 3-point error occurs, the default value of this operand may not allow the processing of some of the back-end servers.

Remarks

The following formula can be used to calculate the default value for this operand:

- When the number of units inside the HA group is 2: *a*
- When the number of units inside the HA group is 3 or larger: $\uparrow (a \times 2) \div (b - 2) \uparrow$

a: *number-of-back-end-servers-inside-host*

b: *number-of-units-inside-HA-group*

When you specify this operand, use the calculation formula (recommended value) shown below to determine the value to be specified for this operand. This calculation formula assumes that when a multi-point error occurs sequentially starting with the unit containing the largest number of units, except the local unit, the remaining units accept the host back-end servers from the failed unit evenly.

$$\text{Recommended value} = \uparrow \sum_{i=1}^{c} a_i \div (b - c) \uparrow$$

a: *number-of-host-back-end*

servers-in-the-unit-ranked-i-th-in-number-of-host-back-end servers-excluding-local-unit

b: *number-of-units-inside-HA-group*

c: *number-of-multi-point-errors-assumed* (< *b*)

62) pd_ha_max_server_process = ***maximum-number-of-user-server-processes-inside-accepting-unit***

~ **<unsigned integer>((1-10000))** ***<<total-value-of-the-pd_max_bes_process-operands-of-host-back-end-servers-inside-unit>>***

Specify this operand when you use the standby-less system switchover (effects distributed) facility. Specify the maximum number of user server processes to be started inside a unit. In the accepting unit, the number of server processes is restricted by the value of the `pd_max_bes_process` operand (the maximum number of processes that can be started) for each host BES and guest BES. In addition, specifying this operand can restrict the total number of server processes inside the unit.

Condition

If you specify this operand, you must also specify the following operands. If they are not specified, the specification of the `pd_ha_max_server_process` operand becomes invalid.

- `pdhagroup`
- `pd_ha_agent = activeunits`

Specification guidelines

Omit this operand if you do not wish to increase the number of processes inside the unit following a system switchover. However, the upper limit of the number of service requests that can be concurrently processed may become restricted. If there is a sufficient amount of resources, specify the combined total value of the maximum number of processes that can be started (value of the `pd_max_bes_process` operand) in the host back-end servers inside the unit and the guest back-end servers that can be accepted. Then, the upper limit of the number of service requests that can be concurrently processed stays the same after a system switchover.

Specifying this operand can prevent an excessive load from being applied to the accepting unit. Determine the value to be specified for this operand by taking into account both the load increase on the unit and the number of service requests that can be concurrently processed following a system switchover.

Note

If the value specified for this operand is smaller than the combined total value of the pd_max_bes_process operand of the host back-end servers inside the unit, that specified value is invalid. In this case, the combined total value of the pd_max_bes_process operand of the host back-end servers inside the unit is assumed (message KFPS05614-W is output).

63) pd_ha_process_count = *number-of-processes-resident-inside-unit-after-acceptance-of-guest-BES*

~ <unsigned integer>((0-10000))

Specify this operand when you use the standby-less system switchover (effects distributed) facility. Specify the combined total of the number of host back-end servers inside the unit and the number of resident processes for the guest back-end servers after these guest back-end servers have been accepted.

You can use the specification for this operand to adjust the number of resident processes in each server after guest back-end servers are accepted.

Condition

If you specify this operand, you must also specify the following operands. If they are not specified, the specification of the pd_ha_max_server_process operand becomes invalid.

- pdhagroup
- pd_ha_agent = activeunits
- pd_process_count operand in the server common definition or back-end server definition

Specification guidelines

- If you omit this operand, HiRDB calculates a value for this operand.
- If some back-end servers cannot start a process, and service requests cannot be processed because a resident process not currently in service is occupying the process, decrease the value of this operand. However, transaction throughput may deteriorate following a system switchover. To improve the transaction throughput following a system switchover, increase the value of this operand. However, if the value of this operand is too large, some of the back-end servers may not be able to process service requests.

Note

If the value specified for this operand is smaller than the combined total value of the pd_process_count operand of the host back-end servers inside the unit, that specified value is invalid. In this case, the combined total value of the pd_process_count operand of the host back-end servers inside the unit is assumed (message `KFPS05615-W` is output).

Remarks

The number of processes resident in each back-end server after guest back-end servers are accepted is the smaller of the following two values:

- Value obtained by proportionately distributing the value of this operand based on the value of the `pd_process_count` operand
- Value of the `pd_process_count` operand of each back-end server

$$\mathrm{MIN}\left\{\begin{array}{l} \texttt{pd\_process\_count}, \\ \downarrow \texttt{pd\_ha\_process\_count} \times \dfrac{\texttt{pd\_process\_count}}{\sum\limits_{\text{Executing BES}} \texttt{pd\_process\_count}} \downarrow \end{array}\right\}$$

The default value of this operand is the value obtained by multiplying the maximum number of user server processes that can be started after guest back-end servers are accepted by the ratio between the number of resident processes and the maximum number of user server processes that can be started before guest back-end servers are accepted. The calculation formula follows.

$$\uparrow \texttt{pd\_ha\_max\_server\_process} \times \dfrac{\sum\limits_{\text{Host BES}} \texttt{pd\_process\_count}}{\sum\limits_{\text{Host BES}} \texttt{pd\_max\_bes\_process}} \uparrow$$

64) pd_ha_resource_act_wait_time = *maximum-wait-time-for-resource-activation*

∼ <unsigned integer>((2-3600)) (seconds)

When you use the standby-less system switchover (effects distributed) facility, this operand specifies the maximum wait time for the running server's resources to be activated at the time of unit startup. Unit startup processing is placed on hold up to the specified amount of wait time. If the resources are activated within the specified amount of time, unit startup processing resumes.

Advantages

When unit startup processing is completed, jobs can be started only if the running server's startup processing is completed in the unit.# By specifying an appropriate value in this operand, you can start your jobs immediately after the unit's startup processing is completed because the unit startup processing will have waited on wait status for resource activation.

#: This event may occur if it takes time to process the command specified in the `usrcommand` operand in the HA monitor's `sysdef` file.

Specification guidelines

Normally, there is no need to specify this operand. You should specify this operand when all the following conditions are applicable:

- The standby-less system switchover (effects distributed) facility is being used.
- The KFPS05623-I message was displayed.
- The target unit for the message contains the running server.

Use the following guideline to determine the value to be specified in this operand:

10 + time required for resource activation processing (seconds)

Time required for resource activation processing: Maximum processing time for the command specified in the usrcommand operand in the HA monitor's sysdef file

Remarks

If the unit does not contain the running server, the running server's startup is placed on hold for the amount of time specified in this operand. However, if all the servers in the unit start as standby servers, unit startup processing is restarted without waiting for the amount of time specified in this operand.

65) pd_ha_ipaddr_inherit = Y | N

This operand specifies whether to inherit IP addresses when using the rapid system switchover facility. If you use the rapid system switchover facility in a HiRDB/Parallel Server, decide whether to specify this operand. How this facility is run differs depending on whether IP addresses are inherited. For details, see the *HiRDB Version 8 System Operation Guide*.

Y: Inherits IP addresses.

N: Does not inherit IP addresses.

Specification guidelines

- Omit this operand if you use the standby-less system switchover (1:1) facility or standby-less system switchover (effects distributed) facility. Even if this operand is specified, it is invalid.
- If you do not use the rapid system switchover facility, you need not specify this operand. Specify the pd_ha_ipaddr_inherit operand of the system common definition.
- If a configuration in which IP addresses are inherited is used for a HiRDB/Parallel Server, specify N (no IP address inheriting) only for the units that are the targets of the rapid system switchover facility

You cannot specify Y for this operand if you specify N for the pd_ha_ipaddr_inherit operand of the system common definition.

3.3.19 Operands related to HiRDB Datareplicator

66) pd_rpl_hdepath = *extracted-side-HiRDB-Datareplicator-directory*

Specifies the extracted side HiRDB Datareplicator directory when the HiRDB Datareplicator linkage facility is used. Specify the name specified in the HDEPATH environment variable of the extracted side HiRDB Datareplicator.

Condition

This specification value is valid when the HiRDB Datareplicator linkage facility is being used and either Y is specified for the pd_rpl_init_start operand or the pdrplstart command is entered.

Note

- To change the extracted side HiRDB Datareplicator directory name, it is necessary to change the HDEPATH environment variable of the extracted side HiRDB Datareplicator.
- When Real Time SAN Replication based on the log-only synchronous method is used, the HiRDB Datareplicator linkage facility cannot be used at the log application site. This operand is ignored, if specified.

Relationship to other operands

If you specify the pd_rpl_reflect_mode operand, make sure that you also specify this operand.

3.3.20 Operands related to communication processing

67) pd_service_port = *port-number-for-high-speed-connection-from-client*

～ <unsigned integer>((5001-65535))

Specifies the port number for high-speed connection. This operand is specified when the high-speed connection facility is used. The following rules apply to the port number that is specified:

- The port number must be unique within the host
- A port number that is different from the port number specified in pd_name_port must be specified

For details about the high-speed connection facility, see the *HiRDB Version 8 Installation and Design Guide.*

Operation method

The port number specified with this operand is specified in the client environment definition (PDSERVICEPORT operand) of the client to which the high-speed connection facility will be applied.

If the mutual system switchover facility is used, a different port number must be specified for each HiRDB instance.

If multiple HiRDB systems or multiple HiRDB units will be started on the same machine, a different port number must be specified for each HiRDB instance.

If a firewall is installed in the HiRDB server, you need to fix the port number for client connection. By specifying this operand, you can fix the port number for client connection. The following client connection method is used in this case:

- If a firewall and NAT are installed in the HiRDB server, use a setting that uses the high-speed connection facility.
- Specify this operand if only a firewall is installed in the HiRDB server. There is no need to specify the client environment definition (PDSERVICEPORT operand).

For details about how to set the HiRDB environment when a firewall or NAT is installed in the HiRDB server, see the *HiRDB Version 8 Installation and Design Guide*.

Notes

- If the port number specified in the pd_name_port operand is specified in pd_service_port, this specification is invalid, and the high-speed connection facility will not be used.
- If the pd_name_port operand is omitted, 20000 must not be specified in pd_service_port.
- Specify a number that is not within the range of port numbers assigned automatically by the OS. The range of port numbers assigned automatically by the OS depends on the OS.
- If a port number is specified that is within the range of port numbers assigned automatically by the OS, that number might be being used by another program. If so, the HiRDB server will not start.
- If a mutual system switchover configuration is used, specify this operand in the unit control information definition. Specify a different port number in the unit control information definition for each unit. If the same port numbers are specified, system switchover will fail in one of the units during a switching attempt.
- If the high-speed connection facility is used to concurrently issue a number of connection requests that exceeds the value specified for the

pd_max_users operand, a shortage occurs in the number of active front-end servers and single servers for extracting connection requests from the message queue. Consequently, connection requests cannot be extracted from the message queue, and the message queue monitoring facility may stop the unit. Therefore, make sure that the number of concurrent connection requests generated does not exceed the value specified for the pd_max_users operand. For details about the message queue monitoring facility, see the *HiRDB Version 8 System Operation Guide*.

68) pd_change_clt_ipaddr = 0 | 1

Specifies the network to be used by the HiRDB server to communicate with HiRDB clients. Normally, you need not specify this operand.

0:

Communication from the HiRDB server to HiRDB clients uses the network in which the IP address specified in the PDCLTRCVADDR operand of the client environment definition is located. If the PDCLTRCVADDR operand is omitted, the IP address of the standard host is assumed.

1:

For communication from the HiRDB server to HiRDB clients, the network used for communication from HiRDB clients to the HiRDB server is used.

69) pd_registered_port =*"port-number-reservation-range"* [,*"port-number-reservation-range"*]...

～ <character string>

In the case of the Linux version, evaluate whether or not this operand should be specified.

When the HiRDB reserved port facility is used, this operand specifies ranges of port numbers to be used for communication by HiRDB.

HiRDB reserved port facility is valid only for server-to-server communication. HiRDB reserved port facility need not be used if the number of ports to be used is small.

Operand specification method

This example provides the following 35,000 port numbers: 6000 to 8999, 12500 to 29999, and 30500 to 44999:

```
set pd_registered_port =
"6000:8999","12500:29999","30500:44999"
```

Advantages

The port numbers to be used for communication between a HiRDB server and HiRDB clients or between HiRDB servers are assigned automatically by the OS. If the communication volume increases significantly, a shortage of port numbers may cause an interruption of processing or may adversely affect the communication processing by other programs. Using this operand, specify a range of port numbers to be used exclusively by HiRDB can prevent such problems.

Specification guidelines

The range of port numbers that can be specified is from 5001 through 49151.

For the number of ports used by HiRDB, see the *HiRDB Version 8 Installation and Design Guide*.

Specification value tuning method

As a guideline, the value obtained from the following formula is the total number of ports that will be needed:

a + b + 100

a: Number of HiRDB reserved ports that are being used (`# OF REGISTERED PORTS`)

b: Number of ports being used that were assigned automatically by the OS because all HiRDB reserved ports were in use (`# OF ASSIGNED PORTS`)

These values can be determined from the statistical information related to system operation obtained by executing the statistics analysis utility.

Operand rules

- Up to 10 ranges of port numbers can be specified.
- If more than one range is specified, ensure that there is no overlap in the port numbers included in the various ranges.
- An ending port number must be greater than the paired starting port number.

Notes

- For the HP-UX, Solaris, and AIX 5L versions, the range of port numbers assigned by the OS is extended automatically. Therefore, a shortage of port numbers will not occur for HiRDB even if this operand is omitted.
- Make sure that the port number reservation ranges specified in this operand do not overlap the port numbers listed below; if there is any overlap, a problem such as a HiRDB startup error may occur:
    - Port numbers specified by the `pd_name_port` operand

    - Port numbers specified by the pd_service_port operand
    - Port numbers specified by the -p option of the pdunit operand
    - Port numbers registered in /etc/services
    - The port number specified by the PDCLTRCVPORT operand in the HiRDB client environment definition (provided that the HiRDB exists on the same server machine)
    - Port numbers being used by other programs
    - Port numbers assigned automatically by the OS
- If multiple HiRDB servers reside on the same server machine (for example, in a multi-HiRDB structure or when the system switchover facility is used), ensure that the port number reservation ranges of the individual HiRDB servers do not overlap.
- In the case of a port that does not support the HiRDB reserved port facility, specification of its port number in this operand is ignored. For the scope of the HiRDB reserved port facility, see the pd_registered_port_level operand.

70) pd_registered_port_check = Y | N | C | W

Specifies whether or not checking is to be performed for overlapping port numbers in the ranges of port numbers specified in the pd_registered_port operand and in the port numbers registered in /etc/services (definition location in an NIS or DNS environment).

Y:

Check for overlap. If an overlap is found, the KFPS00348-E message is output and HiRDB activation is canceled.

N:

Do not check for overlap.

C:

Check for overlap. The HiRDB reserved port facility is not applied to any overlapping port number.

W:

Check for overlap. If an overlap is found, the KFPS00354-W message is output. The HiRDB reserved port facility is not applied to any overlapping port number.

Condition

The pd_registered_port operand must be specified.

Specification guidelines

- If an overlap in port numbers is detected, HiRDB communication may be affected adversely, resulting for example in the receipt of wrong messages or message send failures.
- If Y, C, or W is specified, process server activation in an NIS or DNS environment may slow down.

Remarks

When HiRDB starts, it checks for overlap between the port numbers included in the pd_registered_port operand's range specification and the port numbers specified in the operands listed below. If an overlapping port number is detected, HiRDB displays the KFPS00348-E message and cancels the startup processing.

- pd_name_port operand
- -p option of the pdunit operand
- pd_service_port operand

Note that overlap checking is not performed on port numbers specified in any other operands.

71) pd_registered_port_level = 0 | 1

Specifies the target range for the HiRDB reserved port facility.

Condition

The pd_registered_port operand must be specified.

Specification guidelines

The following shows the target range depending on the value of this operand (0 or 1):

Port numbers used by HiRDB		Value of pd_registered_port_level	
		0	1
Ports used by HiRDB server	Send port number for server-to-server communication	Y	Y
	Receive port number for server-to-server communication	Y	Y
	Port number specified in pd_name_port, pd_service_port, or pdunit -p operand	N	N
	Send port number for command	N	N

Port numbers used by HiRDB		Value of pd_registered_port_level	
		0	1
	Receive port number for command	N	N
	Send port number for client	N	Y
	Receive port number for client	Y	Y
Ports used by HiRDB client	Send port number	N	N
	Receive port number	N	N

Legend:

Y: Subject to the HiRDB reserved port facility. A port number specified in the `pd_registered_port` operand is used.

N: Not subject to the HiRDB reserved port facility.

72) pd_ipc_send_retrycount = *process-to-process-send-retries-count*

~ <unsigned integer>((1-32767)) (Times)

Specifies the number of times process-to-process communication can be attempted. This operand is related to the `pd_ipc_send_retrysleeptime` operand.

Examples

- `pd_ipc_send_retrycount = 500`
- `pd_ipc_send_retrysleeptime = 2`

When the operands are specified in this way, send is attempted up to 500 times and a 2-second sleep occurs between attempts.

Specification guidelines

- Normally, this operand need not be specified.
- Specifying too large a value for this operand increases the CPU lock rate.

73) pd_ipc_send_retrysleeptime = *process-to-process-send-retry-sleep-time*

~ <unsigned integer>((0-60)) (seconds)

Specifies the sleep time between process-to-process send retries.

This operand is related to the `pd_ipc_send_retrycount` operand.

Examples

- `pd_ipc_send_retrycount = 500`
- `pd_ipc_send_retrysleeptime = 2`

When the operands are specified in this way, send is attempted up to 500 times and a 2-second sleep occurs between attempts.

Specification guidelines

- Normally, this operand need not be specified.
- Specifying too large a value for this operand increases communication completion time.

74) pd_ipc_send_count = *server-to-server-send-retries-count*

~ <unsigned integer>((1-32767))

Specifies the number of times a server-to-server send can be performed before the send is completed. A send occurs for up to 5 seconds at each retry. With the default value, send will be performed for up to 60 × 5 seconds = 300 seconds.

Specification guideline

Normally, this operand need not be specified. If transmission timeouts occur frequently, increase this operand's value.

75) pd_ipc_recv_count = *server-to-server-receive-retries-count*

~ <unsigned integer>((1-32767))

Specifies the number of times a server-to-server receive can be performed before receive is completed. Receive occurs for up to 5 seconds at each retry. With the default value, receive is performed for up to 120 × 5 seconds = 10 minutes.

Specification guideline

Normally, this operand need not be specified.

76) pd_ipc_inet_bufsize = *send-receive-buffer-size-for-server-unit-to-unit-communication*

~ <unsigned integer>((0-262144)) (bytes)

Specifies as a multiple of 4096 a size for the send/receive buffer to be used for unit-to-unit communication within a server (TCP NET domain).

Specification guideline

Normally, this operand need not be specified.

Notes

- If `0` is specified, the value set by the OS is used.
- The maximum TCP buffer size depends on the OS.

- The specified buffer size may not be valid depending on the OS being used. In such a case, the value set by the OS is used.

77) pd_ipc_unix_bufsize = send-receive-buffer-size-for-communication-within-server-unit

~ <unsigned integer>((0-262144)) (bytes)

Specifies as a multiple of 4096 a size for the send/receive buffer to be used for communication within a server unit (TCP UNIX domain).

Specification guidelines

Normally, this operand need not be specified.

Notes

- If "0" is specified, the value set by the OS is used.
- The maximum TCP buffer size depends on the OS.
- The specified buffer size may not be valid depending on the OS being used. In such a case, the value set by the OS is used.

78) pd_tcp_inet_bufsize = *send-receive-buffer-size-for-communication-with-HiRDB-client-outside-host-where-HiRDB-server-reside*

~ <unsigned integer>((0-262144)) (bytes)

Specifies as a multiple of 4096 a maximum size for the send/receive buffer to be used for communication with a HiRDB client connected from outside the host where the HiRDB server resides (TCP INET domain).

Specification guidelines

Normally, this operand need not be specified. Specify this operand if one of the following occurs:

- Resending of the data to be sent/received between a HiRDB server and a HiRDB client is observed (specifying this operand can suppress resending)
- The data arrival speed is slower than the data read speed on the receiving side (specifying this operand can prevent the delay caused by waiting for data arrival even when the data volume increases)

Specify for this operand the same value as the buffer size to be used for communication.

Notes

- If 0 is specified, the value set by the OS is used.

- The maximum TCP buffer size depends on the OS.
- The specified buffer size may not be valid depending on the OS being used. In such a case, the value set by the OS is used.

79) pd_tcp_unix_bufsize = ***send-receive-buffer-size-for-communication-with-HiRDB-client-inside-host-where-HiRDB-server-resides***

～ <unsigned integer>((0-262144)) (bytes)

Specifies as a multiple of 4096 a maximum size for the send/receive buffer to be used for communication with a HiRDB client of HiRDB Version 4.0 04-02 or older connected from the host where the HiRDB server resides (TCP UNIX domain).

Specification guidelines

Normally, this operand need not be specified. Specify this operand if one of the following occurs:

- Resending of the data to be sent/received between a HiRDB server and a HiRDB client is observed (specifying this operand can suppress resending)
- The data arrival speed is slower than the data read speed on the receiving side (specifying this operand can prevent the delay caused by waiting for data arrival even when the data volume increases)

Specify for this operand the same value as the buffer size to be used for communication.

Notes

- If 0 is specified, the value set by the OS is used.
- The maximum TCP buffer size depends on the OS.
- The specified buffer size may not be valid depending on the OS being used. In such a case, the value set by the OS is used.

3.3.21 Operands related to Java

Java operands are specified when a Java stored procedure or Java stored function is used. For details about Java stored procedures and Java stored functions, see the *HiRDB Version 8 UAP Development Guide.*

Note that some platforms on which HiRDB can be run do not support Java stored procedures and Java stored functions. For details, see *Environment supporting Java stored procedures and functions* in the *HiRDB Version 8 System Operation Guide*.

80) pd_java_archive_directory = *"JAR-file-storage-directory"*

~ <pathname>

Specifies the name of the directory for storing JAR files used by Java stored procedures or Java stored functions.

Notes

- The specified directory must be created before installation of JAR files.
- The JAR file storage directory is used only for storing JAR files.
- Store only the installed JAR files in the JAR file storage directory.

Operand rules

- Up to 255 characters can be used for the pathname.

81) pd_java_classpath = *"Java-class-path"*

~ <pathname>

Specifies as an absolute pathname the class path to be used by a Java virtual machine.

The class contained in the path specified by this operand can be referenced from the Java method, which is executed as the processing procedure of a Java stored procedure or Java stored function.

If a class with the same name exists in the path specified by this operand and in the JAR file specified as an external routine of the Java stored procedure or Java stored function, the path specified by this operand takes precedence.

Operand rules

- Up to 1024 characters can be used for the pathname.

82) pd_java_runtimepath = *"Java-Runtime-Environment-root-directory*"

~ <pathname>

Specifies as an absolute pathname the root directory of the Java Runtime Environment.

Operand rules

- Up to 255 characters can be used for the pathname.

Notes

Because Java Runtime Environment (JRE) is no longer included, you must add this operand or change its value when you upgrade HiRDB from a version earlier than 07-03 to version 07-03 or later. For notes about upgrading, see *Using Java stored procedures and functions* in the *HiRDB Version 8 Installation and Design Guide*.

83) pd_java_libpath = *"Java-virtual-machine-library-directory"*

~ <pathname>

Specifies the directory that stores the library of the Java virtual machine as a relative pathname to the Java Runtime Environment root directory (the value of the `pd_java_runtimepath` operand).

Operand rules

- Up to 255 characters can be used for the pathname.

Operand specification example

This example assumes that the CPU architecture uses a Hot Spot virtual machine compatible with PA-RISC 2.0; it also assumes use of the HP-UX version of Java Runtime Environment 1.2.2:

```
pd_java_libpath = "lib/PA_RISC2.0/hotspot"
```

Note

Do not specify lib/PA_RISC/classic or lib/PA_RISC2.0/classic for the HP-UX version; if you do, HiRDB may not run correctly.

84) pd_java_stdout_file = *"Java-virtual-machine-standard-output-and-standard-error-output-destination-file"*

~ <pathname>

Specifies as an absolute pathname the file to which the standard output and standard error output are to be output in a Java virtual machine. If this operand is omitted, the standard output and standard error output of the Java virtual machine are ignored.

Specification guidelines

Because the size of the file specified by this operand is extremely large, this operand is not normally specified. It is recommended that this operand be specified during debugging of a Java stored procedure or Java stored function. There is no limit to the size of the file that can be specified by this operand.

Note

If there are simultaneous writing attempts from multiple processes, their output contents cannot be guaranteed.

Operand rules

- Up to 255 characters can be used for the pathname.

3.3.22 Operands related to shared memory

85) SHMMAX *maximum-shared-memory-segment-siz*e

~ **<unsigned integer> (MB)**

- 32-bit mode: **((6-2047))**
- 64-bit mode: **((6-4194304))**

Specifies, in megabytes, the maximum segment size for the shared memory for global buffer.

Specification guidelines

- For this operand, specify a value that is equal to or less than the value specified for the OS's operating system parameter `shmmax` (`shminfo_shmmax` for Solaris, and `SHMMAX` for Linux). If you change the `shmmax` value, also change the value of this operand. For the method of estimating operating system parameters, see the *HiRDB Version 8 Installation and Design Guide*.
- HiRDB allocates shared memory segments for global buffers up to the size specified for this operand. If the total size of the global buffers allocated to RDAREAs inside the server machine exceeds the value specified for this operand, multiple shared memory segments are allocated. If the global buffers are assigned to multiple shared memory segments, the overhead for accessing the shared memory segments increases, causing throughput to decrease. Therefore, try to estimate for this operand a value that enables all of the global buffers to fit inside a single shared memory segment.

  You can use the `pdls -d mem` command to check whether all of the global buffers have been assigned to a single shared memory segment. When there is one server per unit, two pieces of shared memory segment information are displayed. Therefore, when there are multiple servers per unit, all of the global buffers have been assigned to a single shared memory segment if *segment-information-count* - 1 = *server-count* is displayed.
- Allocate at least two shared memory segments (for the unit controller, server, and global buffers) to each unit. You can allocate up to16 shared memory segments.
- When global buffers are dynamically changed, new shared memory segments are allocated and the dynamically changed global buffers are assigned to these new shared memory segments. Use the `pd_max_add_dbbuff_shm_no` operand to specify the maximum number of shared memory segments that can be allocated.

Notes

- When the following condition is satisfied, the required shared memory cannot be allocated, and thus HiRDB cannot be started:

  SHMMAX-value > shmmax-value

- The number of shared memory segments is restricted by the OS's operating system parameters `shmmni` and `shmseg` (`shminfo_shmmin` and `shminfo_shmseg` for Solaris, and `SHMMIN` and `SHMSEG` for Linux). Consequently, when you are allocating multiple shared memory segments for global buffers, you may not be able to allocate shared memory beyond these restrictions. In this case, take one of the following actions:

  - Reduce the number of global buffer sectors.
  - Increase the values of `shmmni` and `shmseg`.
  - Delete unnecessary global buffers.

For details on the shmmax, shmmni, and shmseg parameters, see the applicable OS manual.

Relationship to other operands

This operand is related to the following operands.

- `pd_dbbuff_modify`
- `pdbuffer`
- `pd_max_add_dbbuff_no`
- `pd_max_add_dbbuff_shm_no`

3.3.23 Operands related to sharing of HiRDB system definition files

86) PDCONFPATH *shared-directory-name*

~ **<pathname>**

This operand is applicable to HiRDB/Parallel Server only.

Specifies the name of the directory that is to be shared when sharing of HiRDB system definition files is in effect. Shared files will be created under the specified directory.

For details on how to share HiRDB system definition files, see Section *1.2.8 Sharing of HiRDB system definition files*.

Condition

To use this facility, the file sharing facility (NFS) of Hitachi CSMA/

Configuration Diagram network CD105 (TCP/IP) is required. For details on the file sharing facility (NFS), see the *Hitachi CSMA/Configuration Diagram network CD105 (TCP/IP)*.

Chapter

4. Server Common Definition

This chapter explains the operands of the server common definition.

This chapter contains the following sections:

4.1 Operand formats

The server common definition defines information that is common to individual server definitions (front-end server definitions, dictionary server definition, and back-end server definitions). The values defined here become the default values for operands not specified in the individual server definitions. This section explains the formats used to specify the operands of a server common definition.

Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *4.2 Operands whose specification values can be changed* and *4.3 Operand explanations*.

For users who are creating HiRDB system definitions for the first time

The first step is to determine the values to be specified for the operands shown in boldface type. In principle, HiRDB can be started once the boldface operands have been specified.

No.	Format	Operand category
1	[set pd_max_bes_process = *maximum-number-of-activated-processes-per-back-end-server*]	Processes
2	[set pd_max_dic_process = *maximum-number-of-activated-processes-per-dictionary-server*]	
3	[set pd_process_count = *resident-processes-count* [,*resident-processes-count-at-server-startup*]]	
4	[set pd_server_cleanup_interval = *interval-for-stopping-nonresident-server-processes*]	
5	[set pd_svr_castoff_size = *maximum-memory-size-used-by-server-process*]	
6	[set pd_max_ard_process = *asynchronous-READ-process-count*]	
7	[set pd_dfw_awt_process = *number-of-processes-to-be-written-in-parallel-in-deferred-write*]	

No.	Format	Operand category
8	`[set pd_work_buff_mode = each	pool]`	Work tables
9	`[set pd_work_buff_size =` *work-table-buffer-size*`]`	
10	`[set pd_work_buff_expand_limit =` *work-table-buffer-expansion-limit*`]`	
11	`[set pd_watch_pc_client_time =` *maximum-client-request-wait-time*`]`	System monitoring
12	`[set pd_spd_syncpoint_skip_limit =` *maximum-number-of-skipped-synchronization-point-dumps*`]`	
13	`[set pd_dfw_syncpoint_skip_limit =` *maximum-number-of-skipped-synchronization-point-dumps-resulting-from-delay-of-synchronization-point-dump-acquisition-due-to-deferred-write-processing*`]`	
14	`[set pd_lck_pool_size =` *server-lock-pool-size*`]`	Lock
15	`[set pd_fes_lck_pool_size =` *front-end-server-lock-pool-size*`]`	
16	`[set pd_lck_until_disconnect_cnt =` *total-number-of-tables-and-RDAREAs-to-be-locked-per-server-UNTIL-DISCONNECT-specification*`]`	
17	`[set pd_max_open_holdable_cursors =` *maximum-number-of-holdable-cursors-that-can-be-concurrently-open-when-LOCK-statement-with-UNTIL-DISCONNECT-specification-is-not-executed*`]`	
18	`[set pd_lck_hash_entry =` *lock-pool-hash-entry-count*`]`	
19	`[set pd_dbsync_lck_release_count =` *global-buffer-lock-release-interval-during-synchronization-point-processing*`]`	
20	`[set pd_sql_object_cache_size =` *SQL-object-buffer-size*`]`*	Buffers
21	`[set pd_table_def_cache_size =` *table-definition-information-buffer-size*`]`	
22	`[set pd_auth_cache_size =` *user-privilege-information-buffer-size*`]`	

No.	Format	Operand category
23	`[set pd_view_def_cache_size =` *view-analysis-information-buffer-size*`]`	
24	`[set pd_alias_cache_size =` *table-alias-definition-information-buffer-size*`]`	
25	`[set pd_type_def_cache_size =` *user-defined-type-information-buffer-size*`]`	
26	`[set pd_routine_def_cache_size =` *routine-definition-information-buffer-size*`]`	
27	`[set pd_registry_cache_size =` *registry-information-buffer-size*`]`	
28	`[set pd_sds_shmpool_size =` *single-server-shared-memory-size*`]`	Shared memory
29	`[set pd_dic_shmpool_size =` *dictionary-server-shared-memory-size*`]`	
30	`[set pd_bes_shmpool_size =` *back-end-server-shared-memory-size*`]`	
31	`[set pd_rpc_trace = Y | N]`*	RPC trace information
32	`[set pd_rpc_trace_name = "`*name-for-RPC-trace-collection-files*`"]`*	
33	`[set pd_rpc_trace_size =` *RPC-trace-collection-file-size*`]`*	
34	`[set pd_module_trace_max =` *maximum-number-of-module-traces-that-can-be-stored*`]`*	Troubleshooting information
35	`[set pd_module_trace_timer_level =` *module-trace-output-time-acquisition-method*`]`	
36	`[set pd_max_add_dbbuff_no =` *maximum-global-buffers-count-for-dynamic-addition*`]`	Global buffer
37	`[set pd_max_add_dbbuff_shm_no =` *maximum-shared-memory-segments-count-for-dynamic-addition*`]`	
38	`[set pd_audit_def_buffer_size =` *security-audit-information-buffer-length*`]`	Security audit facility

No.	Format	Operand category
39	`[set pd_java_stdout_file =` `"`*Java-virtual-machine-standard-output-and-standard-error-output-destination-file*`"]`*	Java
40	`[set pd_log_dual = Y | N]`	System log files
41	`[set pd_log_dual_write_method = serial | parallel]`	
42	`[set pd_log_remain_space_check = warn | safe]`	
43	`[set pd_log_singleoperation = Y | N]`	
44	`[set pd_log_rerun_reserved_file_open = Y | N]`	
45	`[set pd_log_rerun_swap = Y | N]`	
46	`[set pd_log_swap_timeout =` *wait-time-for-completion-of-system-log-file-swapping*`]`	
47	`[set pd_log_unload_check = Y | N]`	
48	`[set pd_log_max_data_size =` *log-input/output-buffer-size*`]`	
49	`[set pd_log_write_buff_count =` *log-output-buffer-sectors-count*`]`	
50	`[set pd_log_rec_leng =` *system-log-file-record-length*`]`	
51	`[set pd_spd_dual = Y | N]`	Synchronization point dump files
52	`[set pd_spd_assurance_msg = Y | N]`	
53	`[set pd_spd_assurance_count =` *number-of-guaranteed-valid-generations*`]`	
54	`[set pd_spd_reduced_mode =` *reduced-mode-operation-option*`]`	

No.	Format	Operand category
55	`[set pd_spd_reserved_file_auto_open = Y	N]`	
56	`[set pd_spd_max_data_size =` *synchronization-point-dump-file-buffer-size*`]`	
57	`[set pd_log_sdinterval =` *system-log-output-volume*`[,`*interval*`]]`	
58	`[set pd_sts_initial_error = stop	continue	excontinue]`	Server status files (when an error occurs)
59	`[set pd_sts_singleoperation = stop	continue]`	
60	`[set pd_bes_connection_hold = Y	N]`	BES connection holding facility
61	`[set pd_bes_conn_hold_trn_interval =` *back-end-server-connection-hold-time*`]`	

* If this operand is omitted, the value specified in the same operand in the system common definition or unit control information definition is used.

4.2 Operands whose specification values can be changed

The values of some of the server common definition operands can be changed in the individual server definitions. These operands are indicated below. After a planned termination, forced termination, or abnormal termination of HiRDB, some HiRDB system definition operands can be modified while others cannot be modified. The operands that can be modified are indicated below.

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
1	pd_max_bes_process	—	—	Y	—	—	—	Y	N	Y[1]
2	pd_max_dic_process	—	—	Y	—	—	Y	—	N	Y[1]
3	pd_process_count	—	—	Y	Y	Y	Y	Y	Y[1]	Y[1]
4	pd_server_cleanup_interval	—	—	Y	Y	—	Y	Y	Y	Y
5	pd_svr_castoff_size	—	—	Y	Y	—	Y	Y	Y	Y
6	pd_max_ard_process	—	—	Y	Y	—	Y	Y	Y	Y
7	pd_dfw_awt_process	—	—	Y	Y	—	Y	Y	Y	Y
8	pd_work_buff_mode	—	—	Y	Y	—	Y	Y	Y	Y
9	pd_work_buff_size	—	—	Y	Y	—	Y	Y	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
10	pd_work_buff_expand_limit	—	—	Y	Y	—	Y	Y	Y	Y
11	pd_watch_pc_client_time[3]	—	—	Y	Y	Y	—	—	Y	Y
12	pd_spd_syncpoint_skip_limit	—	—	Y	Y	Y	Y	Y	Y	Y
13	pd_dfw_syncpoint_skip_limit	—	—	Y	Y	—	Y	Y	Y	Y
14	pd_lck_pool_size	—	—	Y	Y	—	Y	Y	Y[1]	Y
15	pd_fes_lck_pool_size	—	—	Y	—	Y	—	—	Y	Y
16	pd_lck_until_disconnect_cnt	—	—	Y	Y	—	Y	Y	N	Y
17	pd_max_open_holdable_cursors	—	—	Y	Y	—	Y	Y	N	N
18	pd_lck_hash_entry	—	—	Y	Y	Y	Y	Y	Y	Y
19	pd_dbsync_lck_release_count	—	—	Y	Y	—	Y	Y	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
20	pd_sql_object_cache_size	Y	—	Y	Y	Y	Y	Y	N	Y
21	pd_table_def_cache_size	—	—	Y	Y	Y	—	—	Y[1,2]	Y[1]
22	pd_auth_cache_size	—	—	Y	Y	Y	—	—	Y[1,2]	Y[1]
23	pd_view_def_cache_size	—	—	Y	Y	Y	—	—	Y[1,2]	Y[1]
24	pd_alias_cache_size	—	—	Y	Y	Y	—	—	Y[1,2]	Y[1]
25	pd_type_def_cache_size	—	—	Y	Y	Y	—	—	Y	Y
26	pd_routine_def_cache_size	—	—	Y	Y	Y	—	—	Y	Y
27	pd_registry_cache_size	—	—	Y	Y	Y	—	—	Y	Y
28	pd_sds_shmpool_size	—	—	Y	Y	—	—	—	Y[1]	Y[1]
29	pd_dic_shmpool_size	—	—	Y	—	—	Y	—	Y[1]	Y[1]
30	pd_bes_shmpool_size	—	—	Y	—	—	—	Y	Y[1]	Y[1]

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
31	`pd_rpc_trace`	Y	Y	Y	Y	Y	Y	Y	Y	Y
32	`pd_rpc_trace_name`	Y	Y	Y	Y	Y	Y	Y	Y	Y
33	`pd_rpc_trace_size`	Y	Y	Y	Y	Y	Y	Y	Y	Y
34	`pd_module_trace_max`	Y	Y	Y	Y	Y	Y	Y	Y	Y
35	`pd_module_trace_timer_level`	Y	Y	Y	Y	Y	Y	Y	Y	Y
36	`pd_max_add_dbbuff_no`	—	—	Y	Y	—	Y	Y	N	Y
37	`pd_max_add_dbbuff_shm_no`	—	—	Y	Y	—	Y	Y	N	Y
38	`pd_audit_def_buffer_size`	—	—	Y	Y	Y	—	—	Y	Y
39	`pd_java_stdout_file`	Y	Y	Y	Y	Y	Y	Y	Y	Y
40	`pd_log_dual`	—	—	Y	Y	Y	Y	Y	N	N
41	`pd_log_dual_write_method`	—	—	Y	Y	Y	Y	Y	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
42	pd_log_remain_space_check	—	—	Y	Y	Y	Y	Y	Y	Y
43	pd_log_singleoperation	—	—	Y	Y	Y	Y	Y	N	N
44	pd_log_rerun_reserved_file_open	—	—	Y	Y	Y	Y	Y	Y	Y
45	pd_log_rerun_swap	—	—	Y	Y	Y	Y	Y	Y	Y
46	pd_log_swap_timeout	—	—	Y	Y	Y	Y	Y	Y	Y
47	pd_log_unload_check	—	—	Y	Y	Y	Y	Y	Y	Y
48	pd_log_max_data_size	—	—	Y	Y	Y	Y	Y	Y[1,2]	Y[1]
49	pd_log_write_buff_count	—	—	Y	Y	Y	Y	Y	Y[2]	Y
50	pd_log_rec_leng	—	—	Y	Y	Y	Y	Y	Y	Y
51	pd_spd_dual	—	—	Y	Y	Y	Y	Y	N	N
52	pd_spd_assurance_msg	—	—	Y	Y	Y	Y	Y	Y	Y

No.	Operand	Definition name							Modifiable after forced or abnormal termination ?	Modifiable after planned termination ?
		SYS	UNT	SVR	SDS	FES	DS	BES		
53	pd_spd_assurance_count	—	—	Y	Y	Y	Y	Y	N	N
54	pd_spd_reduced_mode	—	—	Y	Y	Y	Y	Y	N	Y
55	pd_spd_reserved_file_auto_open	—	—	Y	Y	Y	Y	Y	N	Y
56	pd_spd_max_data_size	—	—	Y	Y	Y	Y	Y	N	N
57	pd_log_sdinterval	—	—	Y	Y	Y	Y	Y	Y	Y
58	pd_sts_initial_error	—	—	Y	Y	Y	Y	Y	Y	Y
59	pd_sts_singleoperation	—	—	Y	Y	Y	Y	Y	Y	Y
60	pd_bes_connection_hold	—	—	Y	—	—	—	Y	Y	Y
61	pd_bes_conn_hold_trn_interval	—	—	Y	—	—	—	Y	Y	Y

Y: Yes, specification value can be modified.

N: No, specification value cannot be modified.

— : Specification value cannot be modified because the operand is not applicable.

SYS: System common definition

UNT: Unit control information definition

SVR: Server common definition

SDS: Single server definition

FES: Front-end server definition

DS: Dictionary server definition

BES: Back-end server definition

[1] If the specified value is too small, it may not be possible to restart HiRDB. When this is the case, restore the value to what it was before the change and then restart HiRDB.

[2] If the specified value is too large, it may not be possible to restart HiRDB. When this is the case, restore the value to what it was before the change and then restart HiRDB.

3 The value for this operand can be different for each client. To change the value for a client, specify the `PDSWATCHTIME` operand in the client environment definition. For details about the `PDSWATCHTIME` operand, see the *HiRDB Version 8 UAP Development Guide*.

4.3 Operand explanations

4.3.1 Operands related to processes

1) pd_max_bes_process = *maximum-number-of-activated-processes-per-back-end-server*

～ <unsigned integer>((1-2048))<<*pd_max_users value*>>

Specifies the maximum number of processes that can be activated per back-end server. For multi front-end servers, processes that exceed the value specified in the `pd_max_users` can sometimes become concentrated in a single back-end server. The `pd_max_bes_process` operand specifies the maximum number of processes that can be activated per back-end server when that number exceeds the value of the `pd_max_users` operand.

Condition

This operand is applicable when multiple front-end servers are used.

Specification guidelines

- The value determined by the following formula indicates the maximum number of processes that may become concentrated in a single back-end server.

  `pd_max_users` value × *number-of-front-end-servers*

  Specify a value for the `pd_max_bes_process` operand by using the value determined here as the upper limit and taking the degree of process concentration in a single back-end server into consideration. Specifying an unnecessarily large value may cause memory shortage.
- If the value specified is smaller than the `pd_max_users` value, the `pd_max_users` value is assumed as the default, and a warning message (`KFPS01888-W`) is output.
- If updatable online reorganization is used, specify an operand value that satisfies the following condition. Otherwise, HiRDB cannot be started.

  `pd_max_bes_process` + `pd_max_reflect_process_count` value ≤ 2048

Tuning the specified value

For details about how to tune the maximum number of processes that can be activated, see the *HiRDB Version 8 System Operation Guide*.

2) pd_max_dic_process = *maximum-number-of-activated-processes-per-dictionary-server*

~ **<unsigned integer>((1-2048))<<*pd_max_users value*>>**

Specifies the maximum number of processes that can be activated per dictionary server. When multiple front-end servers are used, processes that exceed the value specified in `pd_max_users` can sometimes become concentrated in a single dictionary server. Even when multiple front-end servers are not used, processes that exceed the value specified in `pd_max_users` can become concentrated in a single dictionary server if the number of executions of operation commands related to RDAREAs and global buffers (`pdbufls`, `pddbls`, `pdopen`, `pdclose`, `pdhold`, and `pdrels`) exceeds the value specified in `pd_max_users`.

The `pd_max_dic_process` operand specifies the maximum number of processes that can be activated per dictionary server when that number exceeds the value of the `pd_max_users` operand.

Specification guidelines

- The value determined by the following formula indicates the maximum number of processes that may become concentrated in a single dictionary server:

  `pd_max_users` value × *number-of-front-end-servers*

  Specify a value for the `pd_max_dic_process` operand by using the value determined here as the upper limit and taking the degree of process concentration in a single dictionary server into consideration. Specifying an unnecessarily large value may cause memory shortage.

- If the execution of operation commands related to RDAREAs or global buffers causes processes to become concentrated in a single dictionary server, the number of operation commands to be concurrently executed becomes the maximum number for this operand.

- If the value specified is smaller than the `pd_max_users` value, the `pd_max_users` value is assumed as the default, and a warning message (`KFPS01888-W`) is output.

- If updatable online reorganization is used, specify an operand value that satisfies the following condition. Otherwise, HiRDB cannot be started.

  `pd_max_dic_process` + `pd_max_reflect_process_count` value ≤ 2048

Tuning the specified value

For details about how to tune the maximum number of processes that can be activated, see the *HiRDB Version 8 System Operation Guide*.

3) pd_process_count =
***resident-processes-count*[*,resident-processes-count-at-server-startup*]**

〜 <unsigned integer>((0-2048))

resident-processes-count

Specifies the number of processes that can be made resident in each server. A resident process is a process that is activated at the time the server is started.

Advantage

By activating the processes used by transactions that can be processed concurrently by each server at the time of system startup and keeping them resident, the process startup time can be reduced even when new transactions are entered. However, HiRDB startup will take longer.

Specification guidelines

- The value to be specified is determined on the basis of the process private area of each server process and the real memory size of the processor. For details about the process private areas of server processes, see the *HiRDB Version 8 Installation and Design Guide*.
- If a multi front-end server configuration is used and if the `pd_max_bes_process` or `pd_max_dic_process` operand is specified in addition, specify for the `pd_process_count` operand a value that satisfies the following conditions:

  `pd_process_count` value ≤ `pd_max_bes_process` or `pd_max_dic_process` value
- The value specified for this operand should be within the maximum number of processes that can be activated for each server, as shown in the following table:

Server type		Maximum number of processes that can be activated
Single server		`pd_max_users + pd_max_reflect_process_count` value
Front-end server	Multiple front-end servers	`pd_max_users + pd_max_reflect_process_count` value `+ 1`
	Not multiple front-end servers	`pd_max_users + pd_max_reflect_process_count` value
Dictionary server		`pd_max_dic_process`* `+ pd_max_reflect_process_count` value
Back-end server		`pd_max_bes_process`* `+ pd_max_reflect_process_count` value

* If the `pd_max_dic_process` or `pd_max_bes_process` operand is omitted, the

value of the `pd_max_users` operand is assumed as the default.

Tuning the specified value

For details about how to tune the resident processes count, see the *HiRDB Version 8 System Operation Guide*.

Notes

- Because the resident processes count has an effect on memory space availability, specifying an unnecessarily large number may prevent HiRDB from starting.
- If more processes than the resident process count are needed, additional processes are dynamically started, up to the maximum processes count allowed. However, depending on the value specified for the `pd_max_server_process` operand, it may not be possible to start all of the processes indicated by the maximum processes count.

Operand rules

If this operand is omitted (or `0` is specified for it) and the operands are omitted (or `0` is specified for them) in server definitions as well, the following values are assumed:

- For a single server or back-end server: maximum number of processes that can be activated
- For a front-end server or dictionary server: ↑ *maximum-number-of-processes-that-can-be-activated* ÷ 2 ↑

resident-processes-count-at-server-startup

Specifies the number of processes that can be made resident during HiRDB startup.

It is common for resident processes to be activated during HiRDB startup. If there are many resident processes, the amount of time required for HiRDB startup increases proportionately. For example, it takes approximately 1 second to activate a process on a server machine with a 100-MIPS performance rating.

Following are the differences in processing that result depending on whether or not a resident processes count at server startup is specified:

- When there is no specification of a resident processes count at server startup (when, for example, `pd_process_count = 500` is specified)

  All 500 resident processes are activated during HiRDB startup, and HiRDB will not start until they are all activated. In this example, it would take a 100-MIPS server machine about 500 seconds to activate all the resident processes during HiRDB startup.

- When a resident processes count at server startup is specified (when, for example, `pd_process_count = 500,50` is specified)

  Some of the resident processes (50 in this case) are activated during HiRDB startup, and the remaining resident processes are activated after HiRDB startup. HiRDB can be started as long as the specified number of resident processes are activated. In this example, it would take a 100-MIPS server machine about 50 seconds to activate 50 resident processes during HiRDB startup. The remaining 450 resident processes (in this example) would be activated after HiRDB startup.

Advantage

The amount of time required for HiRDB startup is reduced. This option should be used when it is desirable to reduce the HiRDB startup time as much as possible, such as when the system switchover facility is being used.

Specification guidelines

The specified value should be equal to the number of processes that will be required immediately after HiRDB startup is completed.

Note

When a resident processes count at server startup is specified, the value in the `PDCWAITTIME` operand of the client environment definition should be rechecked.

If more UAPs than the resident processes count at server startup are to be executed immediately following HiRDB startup, transaction processing may not be performed until after the remaining resident processes have been activated. Therefore, if the value specified in the `PDCWAITTIME` operand of the client environment definition is small, it may not be possible to process some UAPs due to timeouts. For details about the `PDCWAITTIME` operand, see the *HiRDB Version 8 UAP Development Guide*.

4) pd_server_cleanup_interval = ***interval-for-stopping-nonresident-server-processes***

~ <unsigned integer> ((0-1440))<<0>> (minutes)

Specifies in minutes the interval for checking for nonresident server processes in HiRDB that are to be stopped. The facility for stopping nonresident server processes is applied when the number of executing server processes exceeds the number of processes that can be made resident (value specified by the `pd_process_count` operand). The number of server processes that the facility stops is computed automatically by HiRDB.

Advantage

This facility improves the utilization rates of the memory and of process resources because it increases the number of nonresident processes that can be reused when the workload (number of server processes that are executing) peaks.

Specification guidelines

- For example, if there is only one hour a day during which peak workload occurs and the intervals between peaks within that hour are approximately two minutes apart, `2` should be specified for this operand.
- This facility does not have any effect if the number of server processes that execute concurrently during peak periods is always fewer than the number of resident processes. In this case, this operand should be omitted.

Tuning the specified value

Collect statistical information on system operations for each server for one week. Determine the workload peaks from the number of server processes being serviced (`# OF PROCESSES ON SERVICE`). If that peak value exceeds the currently set number of processes that can be made resident (value specified by the `pd_process_count` operand), determine the interval between individual peaks and set that number of minutes.

However, if there are ample resources, such as memory and CPU, in the server machine, adding the shortfall in the number of processes to the number of resident processes (i.e., increasing the value of the `pd_process_count` operand) would be more effective in improving performance than specifying the `pd_server_cleanup_interval` operand.

Note

When this operand is omitted or `0` is specified, the system checks every 10 seconds for nonresident server processes that are waiting to be serviced and stops any such processes that are found.

5) pd_svr_castoff_size = *maximum-memory-size-used-by-server-process*

～ <unsigned integer>((0-2048))<<0>> (MB)

This operand need not be specified for the Linux version.

Specifies the maximum amount of memory space that can be used by a server process. If at the applicable trigger point shown below the amount of memory being used by a server process exceeds the value specified here, that server process is terminated. When a server process is so terminated, the `KFPS00350-W` message is output. This is called the *facility for monitoring the memory size of* *server processes*; for details, see the *HiRDB Version 8 System Operation Guide*.

Server type	Process name	Process termination trigger
Single server	pdsds	At UAP disconnection
Front-end server	pdfes	At UAP disconnection
Dictionary server	pddic	At transaction decision
Back-end server	pdbes	At transaction decision

Advantages

The facility for monitoring the memory size of server processes resolves the following problem:

- *The memory size of a server-resident process becomes too large during a particular SQL processing, significantly reducing the amount of system memory that is available.*

The HiRDB server releases memory space that is no longer needed. However, even when a program releases memory, the OS holds the memory area itself in the memory management facility inside the applicable process. Consequently, a process that becomes large in terms of using a large memory area even once never shrinks and continues to have an adverse impact on the system, especially in the case of resident processes. The facility for monitoring the memory size of server processes can prevent memory shortages because it terminates even resident processes.

Application criterion

Apply this facility when the amount of memory space used by a HiRDB server process becomes large, resulting in memory shortages.

Specification guidelines

- Maximum value for this operand

  Normally, specify a value by considering the maximum processing capability of HiRDB. Assuming that the maximum number of SQLs may be concurrently connected, determine for this operand a value that satisfies the following condition in each unit:

  $a \times (b + c) < d$

  a: Number of server processes within the unit (*maximum-number-of-concurrent-connections* × *server-count-inside-unit*)

  b: Virtual memory size for one server process immediately following

HiRDB startup

c: `pd_svr_castoff_size` operand value

d: Memory size that can be allocated to a unit (memory size excluding the area used by other programs)

- Minimum value for this operand

  If the value specified for this operand is smaller than the memory size needed for normal SQL processing, the efficiency of making processes resident will deteriorate, resulting in frequent process terminations and restarts. Each time this occurs, a message is output to the syslogfile or the message log file, resulting in further performance degradation. To prevent this, select for this operand a value that satisfies the following condition for each server:

  $a - b < c$

  a: Server process virtual memory size following SQL process termination

  b: Virtual memory size for server process immediately following HiRDB startup

  c: `pd_svr_castoff_size` operand value

The virtual memory size can be determined by an OS command (e.g., the `top` command in HP-UX).

Operand rule

When `0` is specified for this operand, the facility for monitoring the memory size of server processes is not applied.

6) pd_max_ard_process = *asynchronous-READ-process-count*

~ <unsigned integer>((0-256))<<0>>

Specify this operand if you use the asynchronous READ facility. For this operand, specify the number of processes necessary for asynchronous READ operations. For a HiRDB/Parallel Server, this operand specifies the number of processes per server (back-end server or dictionary server). For details on the asynchronous READ facility, see the *HiRDB Version 8 Installation and Design Guide*.

Condition

A value of `1` or greater must be specified for the `-m` option of the `pdbuffer` operand.

Advantage

The asynchronous READ facility is especially effective (improves performance) when character special files, for which input and output take a long time, are used. Conversely, when regular files, for which input and output do not take a long time, or SANRISE disks are used, the asynchronous READ facility may not have much effect because:

- Input/out time does not overlap with CPU time most of the time.
- Communication processing has a large overhead.

Specification guidelines

- Specify `0` or `1`. However, if a value between 2 and 256 is specified for the `-m` option of the `pdbuffer` operand, specify the same value as the `-m` option value. If a value greater than 256 is specified for the `-m` option, specify the same value as the number of disk devices that store RDAREAs and system files (the number of such disk devices per server for a HiRDB/Parallel Server) or `256`.
- Increasing the value of this operand can shorten the processing time when the degree of concurrency is high for the SQL statements to which the asynchronous READ facility is applied. Decreasing the value of this operand may lengthen the processing time when the degree of concurrency is high for the SQL statements to which the asynchronous READ facility is applied. This is because asynchronous READ processes may have to wait for processing completion.
- Because a number of processes equaling *value-of-this-operand* × *server-count* are started, determine a value for this operand by taking resources (shared memory and message queue) into consideration. For the method of estimating shared memory and message queue sizes, see the *HiRDB Version 8 Installation and Design Guide*.

Tuning the specified value

For the method of tuning the specification value (the number of asynchronous READ processes), see the *HiRDB Version 8 System Operation Guide*.

Relationship to other operands

If you change the value of this operand, reevaluate the value of the `pd_max_server_process` operand.

Operand rule

If you specify `0` for this operand, the asynchronous READ facility is not used.

7) pd_dfw_awt_process = ***number-of-processes-to-be-written-in-parallel-in-deferred-write***

~ <unsigned integer>((2-255))

Specify this operand when you use the facility for parallel writes in deferred write processing for all buffer pools. Specify for this operand the number of processes to be processed in parallel. Increasing the number of processes can shorten the write processing time. For details about the facility for parallel writes in deferred write processing, see the *HiRDB Version 8 Installation and Design Guide*.

Specification guidelines

Specify 2, which is the smallest value that enables the facility for parallel writes in deferred write processing. Furthermore, to determine the value for this operand, see *Tuning deferred write processing* in the *HiRDB Version 8 Installation and Design Guide*.

Note

Specifying the facility for parallel writes in deferred write processing increases the number of processes and consequently raises the CPU usage rate.

4.3.2 Operands related to work tables

8) pd_work_buff_mode = each | pool

Specifies the method of allocating buffers when HiRDB creates tables.

each: Allocate a buffer for each work table.

pool: Allocate a buffer pool for each server process.

Specification guidelines

- Normally, pool is specified. pool (default value) is the appropriate specification when a large volume of data is to be retrieved and when manipulations such as join, ORDER BY, and GROUP BY are to be performed.
- When the size of the process private area that can be used for work table buffers is predetermined, pool should be specified. When pool is specified, HiRDB efficiently allocates work table buffers to work tables.

  In such a case, the process private area is occupied on the basis of the value specified in pd_work_buff_size, and input/output operations on work tables are buffered in that pool. Therefore, the process private memory is occupied only to the extent of the value specified in pd_work_buff_size.

9) pd_work_buff_size = *work-table-buffer-size*

~ <unsigned integer> (KB)

- 32-bit mode: **((128-1000000))**
- 64-bit mode: **((128-4000000000))**

Specifies in kilobytes the size of buffers for work tables to be created by HiRDB.

Item	pd_work_buff_mode=each is specified	pd_work_buff_mode=pool is specified
Advantage	A large work table buffer size reduces the number of I/O operations associated with data manipulation, which means that the execution time of SQLs that use work tables is also reduced. However, because each server's process private memory is used, specification of this option should take into account the overall size of the system memory (real memory and virtual memory). If `pd_work_buff_mode` = each is specified, the memory size to be allocated is *value of* `pd_work_buff_size` × *required number of work tables*. A process private memory that has been allocated will remain allocated to that process at least in the virtual memory (will not be released until that process disappears) even after release is requested by HiRDB. Therefore, specifying an unnecessarily large value may cause a virtual memory shortage for other processes.	
Application criterion	`pd_work_buff_mode = pool` should be specified when a large volume of data is to be retrieved and when manipulations such as `join`, `ORDER BY`, and `GROUP BY` are to be performed.	
Specification guidelines	• Specify the size of the buffer to be allocated for one work table. • If a value greater than the work table memory capacity is specified for the work table buffer size, input/output to the work table file during work table creation is eliminated, thus reducing the time necessary for work tables. The following formula can be used to determine the work table memory capacity: `Work table memory capacity = Applicable work table size`* ÷ `2`	• Specify the size of the buffer pool to be allocated for the entire server process. • Specify a value between 4352 and 5120 when a large volume of data is to be retrieved or when manipulations such as join, `ORDER BY`, and `GROUP BY` are to be performed. Specifying such a value increases the unit of sorting input/output, thus reducing the sort time. • If a large value is specified for the work table buffer size, with the total work table memory capacity for each SQL statement as the upper limit, input/output operations on the work table file during work table creation is eliminated, thus reducing the time necessary for work tables. The following formula can be used to determine the total work table memory capacity for each SQL statement: `Total work table memory capacity per SQL statement =` $a \times b + c \times d$

Item	pd_work_buff_mode=each is specified	pd_work_buff_mode=pool is specified
Notes	When multiple users execute processes concurrently or when an SQL statement that uses multiple work tables is executed, the specified size buffer is allocated for each work table. Consequently, specifying a large value may result in a memory shortage.	If the value specified for the work table buffer size is smaller than the number of work tables to be used by each SQL statement, the processing time may become longer than when `each` is specified. Specifically, specify a value that is at least equal to *maximum number of work tables for each SQL statement* × *128*. The following formula can be used to determine the maximum number of work tables for each SQL statement: `Maximum number of work tables for each SQL statement =` *b* `+` *d*
Operand rule	Specify a multiple of 128. If the value is not a multiple of 128, it is rounded up to the next multiple of 128.	• Specify a multiple of 128. If the value is not a multiple of 128, it is rounded up to the next multiple of 128. • Specify at least 384. If a value that is smaller than 384 is specified, it is rounded up to 384.
Default values	• If both this operand and the `pd_work_buff_size` operand of the single server definition are omitted, 128 is assumed. • If both this operand and the `pd_work_buff_size` operand of the back-end server definition and the dictionary server definition are omitted, 512 is assumed.	• If this operand and the `pd_work_buff_size` operand are both omitted in a single server definition, 384 (in the 32-bit mode) or 5120 (in the 64-bit mode) is assumed. • If this operand and the `pd_work_buff_size` operand are both omitted in a back-end server definition or a dictionary server definition, 1024 (in the 32-bit mode) or 5120 (in the 64-bit mode) is assumed.

a:

$\uparrow \{\text{Capacity of work table (for storing column information)}^{*} \text{ (KB)} \div 2\} \div 128 \uparrow \times 128$

b:

Maximum number of work tables (for storing column information)*

c:

$\uparrow \{\text{Capacity of work table (for storing positional information)}^{*} \text{ (KB)} \div 2\} \div 128 \uparrow \times 128$

d:

Maximum number of work tables (for storing positional information)*

* For details about how to determine these values, see the *HiRDB Version 8 Installation and Design Guide*.

10) pd_work_buff_expand_limit = *work-table-buffer-expansion-limit*

~ <unsigned integer> (KB)

- 32-bit mode: **((128-1000000))**
- 64-bit mode: **((128-4000000000))**

The size of the work table buffer to be created by HiRDB is specified by the `pd_work_buff_size` operand. Specify the `pd_work_buff_expand_limit` operand if you want to automatically expand a work table buffer when the space in this buffer becomes insufficient. The work table buffer is expanded up to the size specified by this operand.

For example, when the following values are specified for the operands, a 1,024-KB work table buffer is normally allocated. When this size becomes insufficient, the work table buffer is expanded up to 2,048 KB.

- `pd_work_buff_size = 1024`
- `pd_work_buff_expand_limit = 2048`

HiRDB expands a work table buffer in the following cases:

- The necessary work table buffer cannot be allocated when hash execution is applied to an execution method that uses hash join or subquery hash as the joining method.
- A 128-KB work table buffer allocated to each work table becomes insufficient when multiple work tables are concurrently used.

Condition

The `pd_work_buff_mode` operand must be omitted or `pool` must be specified for it.

Advantage

You can prevent a work table buffer shortage (too small a value specified for the `pd_work_buff_size` operand) from causing UAP errors.

Notes

- A work table buffer is not expanded when either of the following conditions is satisfied:
  - The `pd_work_buff_expand_limit` operand is not specified.
  - `pd_work_buff_expand_limit` operand value ≤

`pd_work_buff_size` operand value

- When a work table buffer is expanded, resident processes are also terminated at the following triggers (except in the Linux version) to free memory:

  HiRDB/Single Server:

  A single server process is terminated when a UAP is disconnected from HiRDB.

  HiRDB/Parallel Server:

  If a holdable cursor is not being used, a back-end server or dictionary server process is terminated when a transaction is normally terminated or cancelled.

  If a holdable cursor is being used, a back-end server or dictionary server process is terminated when a UAP is disconnected from HiRDB.

Operand rule

Specify a multiple of 128. If a value other than a multiple of 128 is specified, it is automatically rounded up to a multiple of 128.

Relationship to other operands

When a work table buffer is expanded for the first time in each server process, the `KFPH29008-I` message is output. Note that you can use the `pd_work_table_option` operand to suppress this message output.

Note

After a work table buffer has been expanded, when the number of work tables being used by the applicable server process goes to zero, the expanded work table buffer is released. The number of work tables being used can go to zero in the following cases:

- All cursors that were being used are closed. (In this case, the number of work tables being used may not go to zero.)
- A transaction is normally terminated or cancelled when a holdable cursor is not being used.
- A UAP is disconnected from HiRDB when a holdable cursor is being used.

Remarks

Hash join, subquery hash execution is applied in the following cases:

- *Application of optimizing mode 2 based on cost* and *hash join, subquery hash execution* are specified in the pd_additional_optimize_level operand, the PDADDITIONALOPTLVL operand of the client environment definition, or the ADD OPTIMIZE LEVEL operand of the SQL compile option.

- HASH is specified for the SQL optimization specification of the joining method inside an SQL statement.
- HASH is specified for the SQL optimization specification of the subquery execution method inside an SQL statement.

4.3.3 Operands related to system monitoring

11) pd_watch_pc_client_time = *maximum-client-request-wait-time*

~ <unsigned integer>((0-65535))<<3600>>(Seconds)

Specifies in seconds the maximum amount of time for a server to wait for the next request from a HiRDB client after the HiRDB server returns a response to a request from a Windows-compatible HiRDB client.

If no request comes from the HiRDB client within the specified amount of time, it will be assumed that an error occurred at the client and the connection between the server and the client will be terminated forcibly. No notice of disconnection is sent to the HiRDB client in such a case.

The time that is monitored is the period between CONNECT and DISCONNECT (i.e., the non-transaction status time), excluding the period between SQL execution startup and COMMIT or ROLLBACK.

Notes

- When 0 is specified, the server waits indefinitely for the next request from the HiRDB client.
- If a small value (e.g., up to 600) is specified for this operand, the HiRDB client may detect *server down* during SQL execution and may be terminated incorrectly.
- For a UNIX version of a HiRDB client (including a Linux version of a HiRDB client), time is not monitored, regardless of the value specified for this operand. To monitor time for a UNIX version of HiRDB client, specify the PDSWATCHTIME client environment definition of the HiRDB client.

Relationship to client environment definition

The value of this operand can be modified for each client. To do so, the PDSWATCHTIME operand must be specified in the client environment definition. For details about the PDSWATCHTIME operand, see the *HiRDB Version 8 UAP Development Guide*.

12) pd_spd_syncpoint_skip_limit = *maximum-number-of-skipped-synchronization-point-dumps*

~ <unsigned integer>((0, 2-100000))

If an event such as an endless loop occurs in a UAP, acquisition of synchronization point dumps may be skipped. If several synchronization point dumps are not acquired (instead, they are skipped), there will be an increase in the number of overwrite-disabled system log files, which may result in a shortage of system log file capacity and ultimately in abnormal termination of the unit.

This operand specifies the maximum number of times synchronization point dumps can be skipped (number of skips per transaction). When the number of skipped synchronization point dumps reaches the specified operand value, the target transaction is cancelled forcibly and rolled back. This is called the *skipped effective synchronization point dump monitoring facility*.

If Real Time SAN Replication based on the log-only synchronous method is used, the skipped effective synchronization point dump monitoring facility cannot be used at the log application site regardless of this operand's value.

For details on this facility, see the *HiRDB Version 8 System Operation Guide*.

Advantage

Specifying this operand provides a means of dealing with endless loops in UAPs.

Specification guideline

Normally, specify `0` for this operand. When `0` is specified, HiRDB computes the upper limit for the skip count. If the specification of `0` causes a problem, change the value of this operand. For the method of estimating the specification value, see the *HiRDB Version 8 System Operation Guide*.

Operand rules

If this operand is omitted, the skipped effective synchronization point dump monitoring facility is not used.

Notes

- The monitoring provided for by the specification of this operand begins the first time a synchronization point dump is collected after HiRDB is started (including a restart).
- When this operand is specified, UAPs that are being executed in the no-log mode are also monitored. If the processing of a UAP being executed in the no-log mode is cancelled, the database cannot be automatically recovered, and thus RDAREAs are placed in the error shutdown state. Therefore, when specifying the upper limit, account

also for the size of the system logs that are output from other transactions inside the server during the processing of UAPs that are executed in the no-log mode.

- The `pdload`, `pdmod`, `pdrorg`, `pdexp`, `pddbst`, `pdgetcst`, `pdrbal`, and `pdvrup` commands are not monitored by this facility.

13) pd_dfw_syncpoint_skip_limit = ***maximum-number-of-skipped-synchronization-point-dumps-resulting-from-delay-of-synchronization-point-dump-acquisition-due-to-deferred-write-processing***

~ <unsigned integer>((0-100000))<<0>>

If a synchronization point occurs within the synchronization point dump acquisition interval but before deferred write processing is completed, acquisition of the synchronization point dump is skipped. This is because acquisition of the synchronization point dump is delayed by the deferred write processing and the number of update buffers output by the synchronization point exceeds the number of update buffers that can be output within the synchronization point dump acquisition interval.

If several synchronization point dumps are skipped, there will be an increase in the number of system log files that cannot be overwritten, which may result in a shortage of system log file capacity and ultimately in abnormal termination of the unit.

This operand specifies the maximum number of times synchronization point dumps can be skipped (number of skips per transaction) due to deferred write processing.

When the number of skipped synchronization point dumps due to deferred write processing reaches the specified operand value, HiRDB determines the maximum number of update buffers in such a manner that acquisition of a synchronization point dump can be completed within the synchronization point dump acquisition interval. If the maximum number of update buffers is exceeded, HiRDB then outputs the oldest update buffer and limits the total size of update buffers at a synchronization point. This is called the *update buffer size restriction facility*.

Advantage

If a synchronization point occurs within the synchronization point dump acquisition interval but before deferred write processing has terminated, the unit terminates abnormally. This operand enables such abnormal termination of the unit to be avoided.

Specification guidelines

Normally, you will omit this operand. If you wish to prevent unit abnormal termination caused by the occurrence of a synchronization point before termination of deferred write processing within the synchronization point dump acquisition interval, specify 1.

If an acceptable number of times synchronization point dumps can be skipped can be determined in advance, such as from the size of the log information, specify that value.

Operand rules

If 0 is specified in this operand, HiRDB does not use the update buffer size restriction facility.

Notes

The following notes explain the period of effectiveness of the update buffer size restriction facility. The period of effectiveness means the interval between issuance of the `KFPH23035-I` message and issuance of the `KFPH23036-I` message.

- If the number of update buffers exceeds the maximum number of update buffers determined by HiRDB, update buffers are output after update processing has executed; this degrades the update processing throughput. You can use the following formula to obtain the maximum number of update buffers determined by HiRDB:

(synchronization point dump interval / unit value of `WRITE`#)
x (1 - (amount of log information from the previous synchronization point dump to the pre-synchronization point)
x (number of buffer sectors in buffer pool / total number of buffer sectors in buffer pool that was updated at the synchronization point)

  #: For details about the unit value of `WRITE`, see the *HiRDB Version 8 System Operation Guide*.

- If the deferred write trigger is specified in the `pd_dbbuff_rate_updpage` or `pdbuffer -y` operand, and each operand value becomes greater than the maximum number of update buffers determined by HiRDB, the maximum number of update buffers determined by HiRDB is changed to the number of update buffers that triggers deferred write processing.

  The value of the `pdbuffer -w` operand is adjusted automatically so that up to the maximum number of update buffers is output for each buffer.

- A skipped synchronization point dump is detected during update buffer output processing at a synchronization point. Therefore, the update buffer size restriction facility may be enabled after a synchronization point dump is skipped and an error message is displayed.

- Normally, there is one output request per synchronization point for the number of processes that used the facility for parallel writes in deferred write processing during synchronization point processing (`pd_dfw_awt_process`).

  However, if the update buffer size restriction facility is used, there would be more than one output request in order to facilitate early detection of skipped synchronization point dumps.

4.3.4 Operands related to lock

14) pd_lck_pool_size = *server-lock-pool-size*

~ **<unsigned integer> (KB)**

- 32-bit mode: **((1-2000000))<<16000>>**
- 64-bit mode: **((1-2000000000))<<32000>>**

For a HiRDB/Single Server, specify in kilobytes the size of the shared memory area to be used for locking by the single server.

For a HiRDB/Parallel Server, specify in kilobytes the size of the shared memory area to be used for locking by back-end servers and dictionary servers.

Specification guidelines

- In the 32-bit mode, the amount of lock pool space required for six lock requests is 1 KB.
- In the 64-bit mode, the amount of lock pool space required for four lock requests is 1 KB.
- The following formulas can be used to determine the value to be specified for this operand:

HiRDB type	Formula
HiRDB/Single Server (32-bit mode)	$\uparrow a \div 6 \uparrow$ (KB)
HiRDB/Parallel Server (32-bit mode)	$\uparrow b \div 6 \uparrow$ (KB)
HiRDB/Single Server (64-bit mode)	$\uparrow a \div 4 \uparrow$ (KB)
HiRDB/Parallel Server (64-bit mode)	$\uparrow b \div 4 \uparrow$ (KB)

a:

Total number of transaction lock requests to be executed concurrently by the single server. The number of lock requests depends on the SQL. For details about how to determine the total number of lock requests, see *E. Determining the Number of Locked Resources*.

b:

Total number of transaction lock requests to be executed concurrently by each server (dictionary server or back-end server). The number of lock requests depends on the SQL. For details about how to determine the total number of lock requests, see *E. Determining the Number of Locked Resources*.

Tuning the specified value

See the usage rate for the locked resources management table (`% OF USE LOCK TABLE`) displayed in the statistical information on system operation by the statistics analysis utility.* If the maximum usage rate equals or exceeds 80%, it is recommended that the operand's value be increased in preparation for future database expansion. If the maximum usage rate does not exceed 10%, it is recommended that the operand's value be decreased to conserve shared memory space.

* For a HiRDB/Parallel Server, see the usage rate for the locked resources management table (`% OF USE LOCK TABLE`) displayed in the statistical information on system operation for each server for the dictionary servers and back-end servers.

Note

If the value specified for this operand is too small, an SQL may return an error.

15) pd_fes_lck_pool_size = *front-end-server-lock-pool-size*

~ <unsigned integer> (KB)

- 32-bit mode: **((1-2000000))**
- 64-bit mode: **((1-2000000000))**

This operand is applicable to HiRDB/Parallel Server only.

Specifies in kilobytes the size of the shared memory area to be used for locking by a front-end server.

Specification guidelines

- In the 32-bit mode, the amount of lock pool space required for six lock requests is 1 KB.
- In the 64-bit mode, the amount of lock pool space required for four lock requests is 1 KB.

- Use the following formula to determine the value of this operand:

  $\uparrow (a + b) \div c \uparrow$ (KB)

  a: Total number of transaction lock requests to be executed concurrently by the front-end server. The number of lock requests depends on the SQL. For details about how to determine the total number of lock requests, see *E. Determining the Number of Locked Resources*.

  b: *(value of* `pd_max_users` + `pd_max_reflect_process_count` + 3*)* × *(value of* `pd_max_access_tables` + 4*)*

  c: Use `6` and `4` for 32- and 64-bit modes, respectively.

Tuning the specified value

See the usage rate for the locked resources management table (`%OF USE LOCK TABLE`) displayed for the front-end server in the statistical information on system operation by the statistics analysis utility. If the maximum usage rate equals or exceeds 80%, it is recommended that the operand's value be increased in preparation for future database expansion. If the maximum usage rate does not exceed 10%, it is recommended that the operand's value be decreased to conserve shared memory space.

Operand default value

The default values of this operand are as follows:

- 32-bit mode

{(`pd_max_users` + `pd_max_reflect_process_count` value + 3) × (`pd_max_access_tables` value + 4)} ÷ 6

- 64-bit mode

{(`pd_max_users` + `pd_max_reflect_process_count` value + 3) × (`pd_max_access_tables` value + 4)} ÷ 4

Note

If the value specified for this operand is too small, an SQL statement may return an error.

16) pd_lck_until_disconnect_cnt = *total-number-of-tables-and-RDAREAs-to-be-locked-until-disconnect-specification*

~ <unsigned integer>((0-140000))<<256>>

Specifies the number of tables and RDAREAs to be held across transactions and the number of resources to be locked for inner replica configuration management and replica group configuration management. According to the value specified for this operand, tables, RDAREAs, and the blocks for which lock with `UNTIL`

DISCONNECT is specified in inner replica configuration management and replica group configuration management are allocated in the shared memory.

Specification guidelines

Normally, this operand need not be specified. Specification of a value other than the default value may be necessary in the following cases:

- When the number of utilities to be executed concurrently increases
- When a holdable cursor is used
- When the local buffer specified in the pdlbuffer operand is used
- When a shared RDAREA is used

For the method of estimating the specification value for this operand, see *D.4 Formula for determining total number of tables and RDAREAs per server locked with UNTIL DISCONNECT specified (pd_lck_until_disconnect_cnt)*.

Tuning the specified value

If the value specified for this operand is small, a transaction may roll back or a utility may terminate abnormally with return code 8. In such cases, the message KFPA11914-E or KFPH28001-E is output. If this occurs, increase the value of this operand.

When the value of this operand is increased, the amount of required memory space increases proportionately. The required memory size can be expressed as follows: *value-of-this-operand* × 48 (64 in the 64-bit mode) bytes.

17) pd_max_open_holdable_cursors = *maximum-number-of-holdable-cursors-that-can-be-concurrently-open-when-LOCK-statement-with-UNTIL-DISCONNECT-specification-is-not-executed*

～ <unsigned integer>((16-1024))<<16>>

When you use holdable cursors for a table for which a LOCK statement with the UNTIL DISCONNECT specification is not executed, this operand specifies the maximum number of holdable cursors that can be concurrently open for each transaction.

Note

Specifying a value other than the default value for this operand increases the amount of shared memory used.

Relationship to other operands

The values specified for this operand and the following operands are used for computing the shared memory size for lock servers. For a 32-bit mode HiRDB system, if the values specified for these operands are too large, the shared memory size of the lock servers exceeds 2 GB, and as a result, HiRDB may not start. Therefore, adjust the values specified for these operands so that the shared memory size of the lock servers does not exceed 2 GB.

- `pd_max_access_tables`
- `pd_max_users`
- `pd_max_bes_process`
- `pd_max_dic_process`
- `pd_lck_hash_entry`
- `pd_lck_pool_size`

For details about shared memory, see the *HiRDB Version 8 Installation and Design Guide*.

18) pd_lck_hash_entry = *lock-pool-hash-entry-count*

~ <unsigned integer>((0-2147483647))<<0>>

Specifies the number of hash table entries to be used in the lock pool. According to the value specified here, HiRDB allocates a lock pool in the shared memory for the unit controller in each server (single server, front-end server, back-end server, and dictionary server).

Specification guidelines

1. Normally, omit this operand.
2. Consider specifying a value for this operand in the following cases:
   - If you do not wish to change the shared memory size if possible when upgrading to Version 06-02 or a newer version, specify `11261`. In this case, the same number of hash entries is allocated as in the older version, the hash table size inside the lock pool remains the same as before.
   - Specifying for this operand a value greater than the recommended value may improve performance. However, specifying a value greater than variable *a* described in *Determining the recommended value* does not improve performance over a case in which *a* is specified.

Operand rules

1. If `0` is specified for this operand or if this operand and the `pd_lck_hash_entry` operand of each server definition are omitted, HiRDB computes a recommended value for each server. For details about recommended values, see *Determining the recommended value* below.
2. If a non-zero value or a non-prime number is specified for this operand,

HiRDB assumes that the largest prime number not exceeding the specified value has been specified.

Note

If the value specified for this operand is too small, hash entry shortage may occur, resulting in performance degradation. If this operand is omitted, neither hash entry shortage nor performance degradation due to hash entry shortage occurs.

Determining the recommended value

The recommended value is described as follows.

Recommended value = Largest prime number not exceeding MAX($\uparrow a \div 10 \uparrow$, 11261)

Variable	Formula for computing the variable		
a	Single server	(`pd_max_users` + `pd_max_reflect_process_count` value + 3) × (`pd_max_access_tables` value + 14) + `pd_lck_pool_size` × *c*	
	Front-end server	If `pd_fes_lck_pool_size` is omitted	(*b* + 3) × (`pd_max_access_tables` value + 4)
		If `pd_fes_lck_pool_size` is specified	`pd_fes_lck_pool_size` value × *c*
	Back-end server or dictionary server	(*b* + 3) × 10 + `pd_lck_pool_size` value × *c*	
b	Front-end server	Multiple front-end server	`pd_max_users` + `pd_max_reflect_process_count` value + 1
		Not multiple front-end server	`pd_max_users` + `pd_max_reflect_process_count` value
	Back-end server	If `pd_max_users` value > `pd_max_bes_process` value	`pd_max_users` + `pd_max_reflect_process_count` value
		If `pd_max_users` value ≤ `pd_max_bes_process` value	`pd_max_bes_process` + `pd_max_reflect_process_count` value
	Dictionary server	If `pd_max_users` value > `pd_max_dic_process` value	`pd_max_users` + `pd_max_reflect_process_count` value
		If `pd_max_users` value ≤ `pd_max_dic_process` value	`pd_max_dic_process` + `pd_max_reflect_process_count` value
c	6 for the 32-bit mode; 4 for the 64-bit mode		

19) pd_dbsync_lck_release_count = *global-buffer-lock-release-interval-during-synchronization-point-processing*

~ <unsigned integer>((0, 100-1073741824))<<10000>>

Specifies the interval for unlocking global buffers, when global buffer locking occurs during synchronization point processing.

During synchronization point processing, search processing occurs on the buffers (update buffers) and must be applied to the disk. Normally, global buffers are unlocked at a specific interval during search processing on the update buffers.

For example, if 100 is specified in this operand, a global buffer is unlocked once when search processing on 100 sectors (global buffer sectors) is completed. After that, the global buffer is locked again and search processing is resumed. In this example, unlocking occurs once every 100 sectors.

Advantage

By specifying this operand, you can adjust the global buffer lock time during synchronization point processing. When a small value is specified in this operand, the global buffer lock time becomes short and transaction performance may improve during synchronization point processing.

To obtain the global buffer pool lock time, execute the statistics analysis utility and in the global buffer pool statistical information check the item called `Buffer pool lock time during synchronization point processing (SYNCL)`.

Specification guidelines

Normally, there is no need to specify this operand. Consider specifying this operand when both the following conditions apply:

- Transaction performance drops during synchronization point processing.
- A large number of buffer sectors is specified in the `-n` option of the `pdbuffer` operand.

Operand rules

- If the specified value is in the range 1 to 99, 100 is set automatically.
- If 0 is specified, global buffers are locked until update buffer search processing is completed.

Notes

If a small value is specified in this operand, the update buffer search time becomes longer due to interruptions by other transactions. The global buffers updated during that time are also output during synchronization point processing. Therefore, the number of update buffers to be output during synchronization point processing increases. To obtain the number of update buffers to be output during synchronization point processing, execute the statistics analysis utility and in the global buffer pool statistical information check the item called `Number of synchronization point output pages (SYNCW).`

4.3.5 Operands related to buffers

20) pd_sql_object_cache_size = *SQL-object-buffer-size*

~ **<unsigned integer> (KB)**

- 32-bit mode: **((22-256000))**
- 64-bit mode: **((22-2000000))**

Specifies in kilobytes the size of the buffer area (shared memory) in which SQL objects are to be placed.

Specification guidelines

- SQL objects are saved in a buffer until the user's transaction has terminated. The buffer must be large enough to store the SQL objects of all transactions that will be executed concurrently.
- SQL analysis can be reduced by saving the SQL objects of static SQLs in the buffer after transaction termination (until the buffer runs out of space) and sharing them among multiple users who execute the same UAP. To effectively utilize the buffer, it should be allocated so that the SQL objects of frequently-used UAPs are resident in the buffer.
- To estimate the buffer size, first determine the buffer size needed for UAP execution from the length of the SQL objects from the SQL statements to be issued by the UAP. Then, compute the buffer size by considering the number of UAPs that will be executed concurrently and the number of concurrently executing users.
- For details about how to estimate the length of the SQL object from a single SQL statement, see *D.2 Formulas for determining size of SQL object buffer (pd_sql_object_cache_size).*

Tuning the specified value

For details about how to tune the SQL object buffer size, see the *HiRDB Version 8 System Operation Guide*.

21) pd_table_def_cache_size = *table-definition-information-buffer-size*

~ **<unsigned integer> (KB)**

- 32-bit mode: **((100-65535))**

- 64-bit mode: **((100-2000000))**
- HiRDB/Single Server

$$《\uparrow \sqrt{\texttt{pd\_max\_users} + 3} \uparrow \times 100》$$

- HiRDB/Parallel Server

$$《\uparrow \sqrt{(\texttt{pd\_max\_bes\_process} \text{ or } \texttt{pd\_max\_dic\_process}\text{, whichever is larger}) + 3} \uparrow \times 100》$$

Specifies in kilobytes the size of the buffer (shared memory) for table definition information that has been used. Table definition information is used during pre-processing of an SQL statement.

Advantages

- Table definition information that has been used is kept in the memory as long as possible and thus can be used subsequently without an input operation.
- Performance improves when a large number of dynamic SQLs are used.
- The number of communications with the dictionary server is reduced (in a HiRDB/Parallel Server).

Specification guidelines

- The total size of the definition information for frequently-used tables is specified.
- For details about how to determine the size of the table definition information buffer per table, see *D.3 Formulas for determining size of table definition information buffer (pd_table_def_cache_size)*.

Tuning the specified value

For details about how to tune the size of the table definition information buffer, see the *HiRDB Version 8 System Operation Guide*.

22) pd_auth_cache_size = *user-privilege-information-buffer-size*

~ <unsigned integer>((1-100))<<1>>(KB)

Specifies in kilobytes the size of the buffer (shared memory) for user privilege information.

Specification guidelines

- The user privilege information buffer stores CONNECT privilege, DBA privilege, and audit privilege information. If this buffer contains no information, information is obtained from a dictionary table during HiRDB connection, thus lengthening the response time. Therefore, specify a buffer size that can store the information for the users who are always connected.

- Storing the user privilege information of each user requires 68 bytes. Use this information when computing the total buffer size.

Tuning the specified value

For details about how to tune the size of the user privilege information buffer, see the *HiRDB Version 8 System Operation Guide*.

23) pd_view_def_cache_size = *view-analysis-information-buffer-size*

~ <unsigned integer> (KB)

- 32-bit mode: **((0-32000))**
- 64-bit mode: **((0-2000000))**

$《\uparrow \sqrt{(\texttt{pd\_max\_users} + 3)} \uparrow \times 8》$

Specifies in kilobytes the size of the buffer (shared memory) for view analysis information.

Advantage

View analysis information that has been used is kept in the shared memory and can be used subsequently without an I/O operation.

Specification guideline

The total size of the view analysis information for frequently-used view tables is specified.

Tuning the specified value

For details about how to tune the size of the buffer for view analysis information, see the *HiRDB Version 8 System Operation Guide*.

24) pd_alias_cache_size = *table-alias-definition-information-buffer-size*

~ <unsigned integer>((0-3000))<<0>> (KB)

Specifies in kilobytes the size of the buffer (shared memory) for table alias definition information.

Specification guidelines

This area should be large enough to hold the information for all frequently used table aliases. The size of the information on each table alias is approximately 164 bytes.

25) pd_type_def_cache_size = *user-defined-type-information-buffer-size*

~ **<unsigned integer><<0>> (KB)**

- 32-bit mode: **((100-65536))**
- 64-bit mode: **((100-2000000))**

This operand is applicable to user-defined types; specification of this operand is recommended when user-defined types will be used.

Specifies in kilobytes the size of the buffer (shared memory) for information on user-defined types. When a user-defined type is used, information on it is stored in this buffer. This information is used during pre-processing of SQL statements.

Advantages

- When user-defined type information is stored in this buffer, it is not necessary to access the dictionary table when the same user-defined type is used subsequently, thus reducing the number of input operations and the CPU usage time.
- The number of communications with the dictionary server is reduced.
- Specification of this operand improves performance when many dynamic SQLs are used.

Specification guidelines

The specified size should be the total of the sizes of the definition information for frequently-used user-defined types that are defined in tables. The following formula can be used to determine the definition information size for one user-defined type:

$\uparrow\{((0.3 + 0.2 \times a + 0.1 \times b) + 3) \div 4\}\uparrow \times 4$ (KB)

a: Number of user-defined type attributes

b: Number of subtypes that have inherited a supertype

Tuning the specified value

For details about how to tune the size of the buffer for user-defined type information, see the *HiRDB Version 8 System Operation Guide*.

26) pd_routine_def_cache_size = *routine-definition-information-buffer-size*

~ **<unsigned integer><<100>> (KB)**

- 32-bit mode: **((0, 20-65536))**
- 64-bit mode: **((0, 20-2000000))**

Specifies in kilobytes the size of the buffer (shared memory) for storing the following types of definition information (this information is used during pre-processing of SQL statements):

- Plug-in facility definition information
- System definition scalar facility definition information
- Routine definition information

Advantages

- When these types of definition information are stored in this buffer, it is not necessary to access the dictionary table when the same information is used subsequently, thus reducing the number of I/O operations and CPU usage time.
- The number of communications with the dictionary server is reduced.

Application criteria

This operand should be specified when a large number of the following types of SQL statements are used:

- SQL statements that use a plug-in
- SQL statements that use the system definition scalar facility
- SQL statements that use a routine

Specification guidelines

For details about how to determine the value for this operand, see *D.5 Formulas for determining size of routine definition information buffer (pd_routine_def_cache_size)*.

Tuning the specified value

For details about how to tune the size of the routine definition information buffer, see the *HiRDB Version 8 System Operation Guide*.

Note

If the value specified for this operand is smaller than the total of the sizes of the definition information for all plug-ins, the definition information on plug-in facilities will not be allocated in the buffer.

27) pd_registry_cache_size = *registry-information-buffer-size*

~ <unsigned integer>((0-65536))<<10>> (KB)

This operand is related to plug-ins. If you use a plug-in that uses registry information, Hitachi recommends the use of this operand. Hitachi also recommends the use of this operand if you use HiRDB Text Search Plug-in.

Specify the size of the buffer (shared memory) for storing registry information (units: KB). When registry information is used, it is stored in the buffer. Registry information is used during the execution of an SQL statement.

Advantages

- Once registry information is stored in this buffer, it is not necessary to access the registry when the same information is used subsequently, thus reducing the number of I/O operations and CPU usage time.
- The number of communications with the dictionary server is reduced.
- Specifying this operand can improve performance when a plug-in that makes frequent use of registry information is used.

Specification guidelines

Use the following formula to determine the registry information buffer size:

$\uparrow (0.3 + a) \uparrow \times b$ (KB)

a: Average registry key length (KB)

The average registry key length can be determined with the following SQL statement:

```
SELECT AVG(KEY_LENGTH) FROM MASTER.SQL_REGISTRY
```

Because the result of this SQL statement is output in bytes, convert it to kilobytes.

b: Number of registry keys registered

The number of registry keys registered can be determined with the following SQL:

```
SELECT COUNT(*)FROM MASTER.SQL_REGISTRY
```

Tuning the specified value

For details about how to tune the size of the buffer for registry information, see the *HiRDB Version 8 System Operation Guide*.

4.3.6 Operands related to shared memory

28) pd_sds_shmpool_size = *single-server-shared-memory-size*

~ <unsigned integer> (KB)

- 32-bit mode: **((1-200000))**

- 64-bit mode: **((1-4000000000))**

This operand applies only to a HiRDB/Single Server, and specifies the size of the area to be used (units: KB) by a single server as part of the shared memory for the unit controller.

Specification guidelines

Normally, omit this operand. HiRDB computes a value for this operand if the pd_sds_shmpool_size operand of the server common definition and single server definition is omitted. HiRDB computes this value based on *Formula for computing the shared memory used by a single server* in the *HiRDB Version 8 Installation and Design Guide*. If you change the value of the operand in the explanation of the variables in this formula, HiRDB automatically re-computes the value for the pd_sds_shmpool_size operand. Note that the value of the pd_assurance_index_no operand is assumed for the *total-number-of-indexes-inside-server* variable inside the formula.

When the pd_assurance_table_no operand is omitted, 100 is assumed for this operand's value.

Tuning the specified value

If any of the following messages is output, increase the specification value of this operand.

- KFPA20003-E
- KFPD00005-E
- KFPD00012-E
- KFPD00021-E
- KFPH20003-E

Notes

- If an unnecessarily large value is specified for this operand, too large a shared memory area will be allocated. If the value specified for this operand is too small, the following may occur:
  - HiRDB will not start
  - A UAP or utility will not execute
- If you omit this operand, you must specify appropriate values for the pd_assurance_table_no and pd_assurance_index_no operands.

29) pd_dic_shmpool_size = *dictionary-server-shared-memory-size*

~ <unsigned integer> (KB)

- 32-bit mode: **((1-200000))**
- 64-bit mode: **((1-4000000000))**

This operand applies only to a HiRDB/Parallel Server, and specifies the size of the area to be used (units: KB) by a dictionary server as part of the shared memory for the unit controller.

Specification guideline

Normally, omit this operand. HiRDB computes a value for this operand if the `pd_dic_shmpool_size` operand of the server common definition and dictionary server definition is omitted. HiRDB computes this value based on *Formula for computing the shared memory used by a dictionary server* in the *HiRDB Version 8 Installation and Design Guide*. If you change the value of the operand in the explanation of the variables in this formula, HiRDB automatically re-computes the value for the `pd_dic_shmpool_size` operand. Note that the number of indexes inside the data dictionary table is assumed for the *total-number-of-indexes-inside-server* variable inside the formula. In the formula, for the variables *Number of global buffer pools for index* and *Total number of global buffers (number of pdbuffer operands)*, 500 is assumed in the 32-bit mode and 1000 is assumed in the 64-bit mode.

Tuning the specified value

If any of the following messages is output, increase the specification value of this operand.

- `KFPA20003-E`
- `KFPD00005-E`
- `KFPD00012-E`
- `KFPD00021-E`
- `KFPH20003-E`

Notes

- If an unnecessarily large value is specified for this operand, too large a shared memory area will be allocated. If the value specified for this operand is too small, the following may occur:
  - Unit will not activate
  - A UAP or utility will not execute
- If you omit this operand, make sure that you specify an appropriate value in the `pd_assurance_index_no` operand.

30) pd_bes_shmpool_size = *back-end-server-shared-memory-size*

~ <unsigned integer> (KB)

- 32-bit mode: **((1-200000))**
- 64-bit mode: **((1-4000000000))**

This operand applies only to a HiRDB/Parallel Server, and specifies the size of the area to be used (units: KB) by a back-end server as part of the shared memory for the unit controller.

Specification guidelines

Normally, omit this operand. HiRDB computes a value for this operand if the `pd_bes_shmpool_size` operand of the server common definition and back-end server definition is omitted. HiRDB computes this value based on *Formula for computing the shared memory used by a back-end server* in the *HiRDB Version 8 Installation and Design Guide*. If you change the value of the operand in the explanation of the variables in this formula, HiRDB automatically re-computes the value for the `pd_bes_shmpool_size` operand. Note that the value of the `pd_assurance_index_no` operand is assumed for the *total-number-of-indexes-inside-server* variable inside the formula.

In the formula, for the variables *Number of global buffer pools for index* and *Total number of global buffers (number of pdbuffer operands)*, 500 is assumed in the 32-bit mode and 1000 is assumed in the 64-bit mode.

Tuning the specified value

If any of the following messages is output, increase the specification value of this operand.

- `KFPA20003-E`
- `KFPD00005-E`
- `KFPD00012-E`
- `KFPD00021-E`
- `KFPH20003-E`

Notes

If an unnecessarily large value is specified for this operand, too large a shared memory area will be allocated. If the value specified for this operand is too small, the following may occur:

- Unit will not activate
- A UAP or utility will not execute

If you omit this operand, you must specify appropriate values for the `pd_assurance_table_no` and `pd_assurance_index_no` operands.

4.3.7 Operands related to RPC trace information

31) pd_rpc_trace = Y | N

Specifies whether or not RPC trace information is to be collected. HiRDB maintenance information is output in the RPC trace information.

Normally, this operand should be omitted.

`Y`: Collect RPC trace information.

`N`: Do not collect RPC trace information.

Note

Specifying `Y` for this operand degrades communication performance.

32) pd_rpc_trace_name = "*name-of-RPC-trace-collection-file*"

~ <pathname of up to 254 characters>

Specifies an `absolute pathname` for the filename for the RPC trace files. Three RPC trace files are created, with 1, 2, and *l* suffixed to the specified file name.

Note

Files with a maximum size of `pd_rpc_trace_size` *value* × 2 are created under the directory specified by this operand. Attention should be paid to the file capacity.

33) pd_rpc_trace_size = *RPC-trace-collection-file-size*

~ <unsigned integer>((1024-2147483648)) (Bytes)

Specifies in bytes the size of the RPC trace files.

The value specified by this operand determines the size of both RPC trace file 1 and RPC trace file 2.

Specification guidelines

An RPC trace collects in a single file in time order the information on the communications among all processes. If the volume of this information exceeds the specified value, trace data output is switched to the other file, which means that older trace information will be overwritten. When this happens, the amount of trace information that is available may be inadequate, making troubleshooting difficult. For this reason, at least 1,000,000 should be specified for this operand.

Note

The value specified by this operand does not apply to RPC trace file *l*, because its size is fixed at 0 bytes.

4.3.8 Operands related to troubleshooting information

34) pd_module_trace_max = *maximum-number-of-module-traces-that-can-be-stored*

~ <unsigned integer>((126-16383))

A HiRDB process records the history of the executed functions and macros inside the process private memory. This history is called a *module trace*. This operand specifies the number of module trace records. The content of this history is loaded into the `core` file and is output when a process error occurs.

Specification guidelines

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

Process private memory of the following size is allocated to each process:

In the 32-bit mode: 64 + 48 × `pd_module_trace_max` *operand value* (bytes)

In the 64-bit mode: 64 + 64 × `pd_module_trace_max` *operand value* (bytes)

35) pd_module_trace_timer_level = 0 | 10 | 20

Specifies how to acquire the time to be output in module traces. The following table explains the meaning of the value specified for this operand.

Specified value	Time acquisition method
`0`	Time is output in seconds at every module trace output location.
`10`	Time is output in microseconds only at performance-critical module trace output locations, such as those before and after input/output processing, and time is output in seconds at other locations.
`20`	Time is output in microseconds at every module trace output location.

Specification guidelines

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

If you specify a value other than 0 for this operand, a function for acquiring time in microseconds is issued, and as a result, system performance may decline.

4.3.9 Operands related to global buffers

36) pd_max_add_dbbuff_no = *maximum-global-buffers-count-for-dynamic-addition*

～<unsigned integer>((1-32752))

In order to change global buffers dynamically, this operand specifies the maximum number of global buffers (per server) that can be added dynamically by the pdbufmod command.

Condition

Y must be specified in the pd_dbbuff_modify operand.

Default value

This operand's default value is as follows:

Condition		Default value
32-bit mode	*a* ≥ 500	256
	a < 500	500 - *a*
64-bit mode	*a* ≥ 1000	256
	a < 1000	1000 - *a*

a: Number of global buffers allocated per server during HiRDB startup

Specification guidelines

- Estimate the number of global buffers to be added dynamically by the pdbufmod command and then specify a sufficient value based on that value.
- Determine the operand's value in such a manner that the following condition is satisfied:

  Value of pd_max_add_dbbuff_no ≤ 2000000 - number of global buffers allocated per server during HiRDB startup

Notes

Do not specify an unnecessarily large value in this operand. If this operand's value is too large, the shared memory used by HiRDB increases, which may result in a shortage of shared memory and an inability of HiRDB to start.

Relationship to other operands

This operand is related to the following operands:

- `SHMMAX`
- `pdbuffer`
- `pd_max_add_dbbuff_shm_no`

37) pd_max_add_dbbuff_shm_no = *maximum-shared-memory-segments-count-for-dynamic-addition*

∼ <unsigned integer>((1-32752))

In order to change global buffers dynamically, specifies the maximum number of shared memory segments (per server) that can be allocated when dynamic addition is performed by the `pdbufmod` command.

Condition

`Y` must be specified in the `pd_dbbuff_modify` operand.

Default value

This operand's default value is as follows:

Condition		Default value
`pd_max_add_dbbuff_no` operand is omitted	32-bit mode	(16 - *a*) + 500
	64-bit mode	(16 - *a*) + 1000
`pd_max_add_dbbuff_no` operand is specified	32-bit mode	Value of ↓`pd_max_add_dbbuff_no` x 1.5 + 16↓ (If the value is 32752 or greater, 32752 is set)
	64-bit mode	

a: Number of shared memory segments allocated per server during HiRDB startup

Specification guidelines

Estimate the number of global buffers to be added dynamically by the `pdbufmod` command and then specify an appropriate value.

Notes

- If the following condition is satisfied, the value of the `pd_max_add_dbbuff_no` operand is assumed in this operand:

  Value of `pd_max_add_dbbuff_shm_no` < value of `pd_max_add_dbbuff_no`

The value of the `pd_max_add_dbbuff_no` operand is also assumed when the result obtained from the formula shown in *Default value* above satisfies the above condition.

- Do not specify an unnecessarily large value in this operand. If this operand's value is too large, the shared memory used by HiRDB increases, which may result in a shortage of shared memory and an inability of HiRDB to start.
- If the size of a shared memory segment to be added exceeds the `SHMMAX` operand value, shared memory is divided into multiple segments based on the `SHMMAX` operand value as the maximum value. Either increase the `SHMMAX` operand value based on the size of the shared memory segment to be added or increase the `pd_max_add_dbbuff_no` operand value so that no shortage occurs when the shared memory is segmented.
- If you have changed the value of this operand, re-evaluate the maximum shared memory segment size in the OS parameters, the maximum number of shared memory segments in the system, and the maximum number of shared memory segments per process.

Relationship to other operands

This operand is related to the following operands:

- `SHMMAX`
- `pdbuffer`
- `pd_max_add_dbbuff_no`

4.3.10 Operands related to the security audit facility

38) pd_audit_def_buffer_size = *security-audit-information-buffer-length*

~ <unsigned integer>((1-2000000)) (kilobytes)

Specifies (in kilobytes) the buffer size (shared memory) used for storing information for the security audit facility.

Specification guidelines

Use the following formula to determine the security audit information buffer length:

$\uparrow 0.3 + a \times 0.25 \uparrow$ (kilobytes)

a: Number of objects for the narrowing condition that was specified in the audit trail of the security audit facility

Default value

If this operand is omitted, the following buffer size is acquired during HiRDB startup; if this amount of memory cannot be allocated, HiRDB will not start:

$$\uparrow 0.3 + \max\{(a + 100),(a \times 1.2)\} \times 0.25 \uparrow$$

(kilobytes)

a: Number of objects for the narrowing condition that was specified in the audit trail of the security audit facility

Notes

If the amount of memory required by this operand's specification cannot be allocated, HiRDB will not start.

4.3.11 Operands related to Java

Java operands are specified when a Java stored procedure or Java stored function is used. For details about Java stored procedures and Java stored functions, see the *HiRDB Version 8 UAP Development Guide.*

Note that some platforms on which HiRDB is run do not support Java stored procedures or Java stored functions. For details, see *Environment supporting Java stored procedures and functions* in the manual *HiRDB Version 8 System Operation Guide*.

39) pd_java_stdout_file = *"Java-virtual-machine-standard-output-and-standard-error-output-destination-file"*

~ **<pathname>**

Specifies as an absolute pathname the file to which the standard output and standard error output are to be output in a Java virtual machine. If this operand is omitted, the standard output and standard error output of the Java virtual machine are ignored.

Specification guideline

Because the size of the file specified by this operand is extremely large, this operand is not normally specified. It is recommended that this operand be specified during debugging of a Java stored procedure or Java stored function. There is no limit to the size of the file that can be specified by this operand.

Note

If there are simultaneous writing attempts from multiple processes, their output contents cannot be guaranteed.

Operand rules

- Up to 255 characters can be used for the pathname.

4.3.12 Operands related to system log files

40) pd_log_dual = Y | <u>N</u>

Specifies whether or not dual system log files are to be used.

`Y`: Use dual system log files.

`N`: Do not use dual system log files.

Advantages

When dual system log files are used, HiRDB collects the same system log information in both files, which are called File A and File B. If a failure occurs in one of the files while the collected system log file is being loaded, the file can be loaded from the other file, resulting in higher system reliability.

Relationship to other operands

When use of dual system log files is specified, the name of the File B system log file must be specified with the `pdlogadpf` operand in each server definition.

41) pd_log_dual_write_method = <u>serial</u> | parallel

This operand is applicable only to the AIX 5L version of HiRDB.

This operand specifies whether to use the facility for parallel output of system logs. For details about this facility, see the *HiRDB Version 8 Installation and Design Guide*.

`serial`: The facility for parallel output of system logs is not used.

`parallel`: The facility for parallel output of system logs is used.

When `parallel` is specified for this operand, HiRDB uses the Asynchronous I/O Subsystem to execute parallel output to the system log files.

Conditions

When you specify `parallel` for this operand, the following conditions must be satisfied:

- `pd_log_dual = Y`
- The system log files are located in character special files.

If either of these conditions is not satisfied, the facility for parallel output of system logs is not used regardless of the value specified for this operand.

Additionally, when you specify `parallel` for this operand, the

Asynchronous I/O Subsystem must be installed.

42) pd_log_remain_space_check = warn | safe

Specifies the processing to be performed by HiRDB when the available space in the system log file falls below the warning level. This is called the facility for monitoring the free area for the system log file. For details on this facility, see the *HiRDB Version 8 System Operation Guide*.

`warn`:

When the available space in the system log file falls below the warning level, the `KFPS01162-W` message is output.

`safe`:

When the available space in the system log file falls below the warning level, scheduling of new transactions is suppressed, all transactions inside the server are forcibly terminated, and the `KFPS01160`-E message is output.

Specification guideline

Hitachi recommends the specification of `safe` because it can reduce the probability of abnormal termination of units due to system log file space shortage. However, when `safe` is specified, all transactions inside the server are forcibly terminated when a shortage occurs in the available space in the system log file. Therefore, the design of the system log file requires more accuracy. For details on system log file design, see the *HiRDB Version 8 Installation and Design Guide*.

43) pd_log_singleoperation = Y | N

This operand is applicable when dual system log files are used; it need not be specified when dual system log files are not used.

Specifies whether or not the single-operation mode is to be used for the system log files. Even if an error occurs in a system log file, making no dual system log files available, HiRDB (or a unit for a HiRDB/Parallel Server) can continue processing using the remaining single normal system log file without being abnormally terminated. This is called *single operation of the system log files*.

The mode in which continuation of processing is permitted only with both system log files available (normal operation mode) is called *double operation of the system log files*.

`Y`: Use single operation of the system log files.

`N`: Do not use single operation of the system log files. Both system log files must always be used.

Condition

Y must be specified in the pd_log_dual operand.

44) pd_log_rerun_reserved_file_open = Y | N

Specifies whether or not a system log file is to be opened automatically.

If no system log file that can be overwritten is available during a HiRDB (or unit) restart, HiRDB opens a reserved file (if one is available), makes it overwritable, and continues processing. This is called *automatic opening of system log file*.

A reserved file is used in the following cases:

- Between the time of a restart and the time when the first synchronization point dump is collected
- When none of the opened file groups can be overwritten

Y: Open a system log file automatically (open and use a reserved file).

N: Do not open a system log file automatically (do not use a reserved file).

Advantages

When Y is specified, the unit can be restarted as long as a reserved file is available, even though no file that can be swapped in is available during the unit restart.

However, if a file that is in unload wait status is available, the unit is stopped; once that file has been unloaded, the unit can be restarted.

45) pd_log_rerun_swap = Y | N

Specifies whether or not the system log files are to be swapped during a HiRDB (or unit) restart.

Y: Swap the system log files.

N: Do not swap the system log files.

Advantage

When Y is specified, there can be a physical separation between the system log files used before and after a restart. Therefore, the system log file that was being used before the restart can be reused during server operation.

46) pd_log_swap_timeout = *wait-time-for-completion-of-system-log-file-swapping*

~ <unsigned integer>((1-32580))<<180>> (seconds)

Specifies in seconds the wait time during which swapping of system log files must be completed. If a swap of system log files is not completed within the specified amount of time, the unit terminates abnormally.

Specification guidelines

Normally, there is no need to specify this operand. If it takes a long time to swap system log files for a reason such as poor machine performance, you can specify this operand with a value that is greater than the default value. To detect errors or delays quickly during system log file swapping so that the unit can be terminated (such as because of a disk failure), reduce the value of this operand.

47) pd_log_unload_check = Y | N

Specifies whether or not HiRDB is to check the unload status of system log files.

`Y`:

Check the unload status (normal operation).

`N`:

Do not check the unload status. A system log file is placed in swappable status, regardless of its unload status, when both of the following conditions are satisfied:

- It is in overwritable status
- It is in extraction completed status (HiRDB Datareplicator)
- Overwriting permitted status for online reorganization (HiRDB Staticizer Option)

In such a case, the system log file operation method is to release checking of unload status. For details about this operation method, see the *HiRDB Version 8 System Operation Guide*.

Advantages

Specifying `N` provides the following advantages:

- Operations are simplified because the system log file unload operation is eliminated.
- It is not necessary to provide files for storing unload files.

Specification guideline

`N` should be specified if the system log file will not be needed for database recovery (i.e., if recovery from a backup collection point will be sufficient).

Notes

The following points apply when `N` is specified:

- Database can be recovered only if backups have been made.
- If this option is specified when the system log file is required for database recovery, it will not be possible to recover the database.

48) pd_log_max_data_size = *log-input/output-buffer-size*

～ <unsigned integer>((32000-523000))<<400000>> (Bytes)

Specifies in bytes the size of the buffer to be used for system log input/output operations.

Specification guideline

- HiRDB/Single Server

  Change the specification value according to the tuning method.

- HiRDB/Parallel Server

  Specify a value that satisfies conditional expression 1 below. If `uap` is specified in the `pd_rpl_reflect_mode` operand or a recovery-unnecessary front-end server is used, specify a value that satisfies both the conditional expressions below (1 and 2). To optimize the value, use the tuning method for the specification value.

  Conditional expression 1: log input/output buffer length ≥ *a*

  a: 72 x total number of back-end servers + 1432

  Conditional expression 2: log input/output buffer length ≥ *b*

  b: (Maximum number of back-end servers subject to update processing by a single transaction + 1) x 128 + 64

Tuning the specified value

A value (other than the default value) may need to be specified for this operand after the following types of statistics analysis utility statistical information related to system operation have been checked:

- Number of buffer sectors waiting for input/output (`# OF BUFFER FOR WAIT I/O`)
- Number of waits caused by lack of a current buffer (`# OF WAIT THREAD`)

  If the number of waits caused by lack of a current buffer is not `0`, increase the value of this operand.

Relationship to other operands

- Use this operand and the `pd_log_write_buff_count` operand to determine the log output buffer size.
- If `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand, the default value for this operand is `32000`.

49) pd_log_write_buff_count = *log-output-buffer-sectors-count*

~ <unsigned integer>((3-65000))<<10>>

Specifies the number of buffer sectors to be used for system log output.

Tuning the specified value

A value (other than the default value) may need to be specified for this operand after the following types of statistics analysis utility statistical information related to system operation have been checked:

- Number of times the buffer was full (`# OF BUFFER FULL`)
- Number of waits caused by lack of a current buffer (`# OF WAIT THREAD`)

If either of these values is large, a larger value should be specified in order to improve throughput.

Relationship to other operands

- Use this operand and the `pd_log_max_data_size` operand to determine the log output buffer size.
- If `v6compatible` or `v7compatible` is specified in the `pd_sysdef_default_option` operand, the default value for this operand is `3`.

50) pd_log_rec_leng = *system-log-file-record-length*

~ <unsigned integer>((1024, 2048, 4096))<<4096>> (Bytes)

Specifies the record length for the system log files; the specifiable values are 1024, 2048, and 4096.

The record length specified in the `-l` option of the `pdloginit` command should be specified for this operand.

Notes

- If a value that is different from the record length specified by the `-l` option of the `pdloginit` command is specified for this operand, system log files cannot be opened.
- For details about how to modify the system log file record length, see the *HiRDB Version 8 System Operation Guide*.

4.3.13 Operands related to synchronization point dump files

51) pd_spd_dual = Y | <u>N</u>

Specifies whether to use dual synchronization point dump files.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for the server are also used at the log application site.

Y: Uses dual synchronization point dump files.

N: Does not use dual synchronization point dump files.

Advantage

When dual synchronization point dump files are used, HiRDB collects the same synchronization point dump in both dump files A and B. Even when an error occurs in one of the files when the collected synchronization point dump is being loaded, the synchronization point dump can be loaded from the other file, thus improving system reliability.

Relationship to other operands

To use dual synchronization point dump files, specify the name of synchronization point dump file B in the pdlogadpf operand of each server definition.

52) pd_spd_assurance_msg = Y | N

Specifies whether or not the KFPS02183-I message is to be output when a synchronization point dump is completed.

Y: Output the message.

N: Do not output the message.

53) pd_spd_assurance_count = *number-of-guaranteed-valid-generations*

~ <unsigned integer>((1-2))<<1>>

Specifies the range of system log files to be saved during HiRDB operation as a protection against events such as a synchronization point dump file input error during system recovery. What is specified here is a number of synchronization point dump file generations; the specified value is referred to as the *number of guaranteed-valid generations*. The number of past generations of synchronization point dump files specified here are overwrite disabled.

If Real Time SAN Replication based on the log-only synchronous method is used, this operand is ignored, if specified, at the log application site. The system log files used at the log application site are overwritten by the transaction execution site even if they include system log information that has not been applied.

Advantage

When 2 is specified as number of guaranteed-valid generations and an error occurs in the most recent synchronization point dump file generation, the system can be recovered using the preceding synchronization point dump file generation, resulting in higher reliability.

Specification guideline

- To improve reliability, `2` should be specified for the number of guaranteed valid generations. However, the number of overwrite disabled synchronization point dump files increases (to two).
- If reliability has been improved with the use of dual synchronization point dump files, Hitachi recommends that you omit this operand or specify `1` for it.

Notes

- The minimum number of synchronization point dump files needed is *number-of-guaranteed-valid-generations* + 1.
- Specifying `2` increases the number of overwrite disabled synchronization point dump files. Moreover, the system log files that correspond to the overwrite disabled synchronization point dump files are also overwrite disabled. Consequently, specifying `2` for the number of guaranteed valid generations increases the number of overwrite disabled system log files. As a result, a shortage may occur in the number of system log files that can be swapped in. To prevent this, it may be necessary to reevaluate the system log file capacity.

54) pd_spd_reduced_mode = *reduced-mode-operation-option*

~ <unsigned integer>((0-2))<<0>>

Specifies whether or not the reduced mode operation for synchronization point dump files is to be used.

Reduced mode operation is a facility for continuing processing if at least two synchronization point dump files can be used, even if an event such as a file error during HiRDB operation or restart reduces the number of synchronization point dump files to or below the number of required files (*number of guaranteed valid generations*[*] + 1).

[*] Value specified for the `pd_spd_assurance_count` operand.

`0`: Do not use the reduced mode operation.

`1`: Use the reduced mode operation.

`2`: Use the reduced mode operation and issue a warning message whenever a synchronization point dump is being acquired during the reduced mode operation.

55) pd_spd_reserved_file_auto_open = Y | <u>N</u>

Specifies whether or not a synchronization point dump file is to be opened automatically. When `Y` is specified and an error in a synchronization point dump file reduces the number of synchronization point dump files to the number of files necessary for operation (*number of guaranteed valid generations*[*] + 1), HiRDB opens a reserved file (if one is available), makes it overwritable, and continues processing. This is called *automatic opening of synchronization point dump file*.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for the server are also used at the log application site.

[*] Value specified for the `pd_spd_assurance_count` operand.

`Y`:

Open a synchronization point dump file automatically. A reserved file is opened when the number of files falls below the number necessary for operation (*number of guaranteed valid generations* + 1).

`N`:

Do not open a synchronization point dump file. No reserved file is opened, even though the number of files falls below the number necessary for operation (*number of guaranteed valid generations* + 1).

Relationship to other operands

This operand has a higher priority than the `pd_spd_reduced_mode` operand.

56) pd_spd_max_data_size = *synchronization-point-dump-file-buffer-size*

~ <unsigned integer>((32000-4000000))<<32768>> (Bytes)

Specifies in bytes the size of the buffer (shared memory) to be used for synchronization point dump file input/output operations.

The value specified here affects the number of synchronization point dump file input/output operations.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for the server are also used at the log application site.

Specification guidelines

- Normally, this operand need not be specified.
- When the value of this operand is increased, the number of synchronization point dump file input/output operations is reduced.

57) pd_log_sdinterval = *system-log-output-volume*[,interval]

Specifies the collection interval for synchronization point dumps. The following should be taken into consideration in specifying this operand:

- Volume of system log information output since the previous synchronization point
- Amount of time that has elapsed since the previous synchronization point

***system-log-output-volume*: ~ <unsigned integer>((100-100000))<<5000>> (Number of log blocks)**

Specifies an interval between synchronization point dumps in terms of the number of blocks of log information. A synchronization point dump is collected each time system log information equivalent to the number of log blocks specified here has been output.

***interval*: ~ <unsigned integer>((0 or 10-1440))<<60>> (Minutes)**

Specifies a synchronization point dump collection interval in terms of number of minutes between synchronization point dumps.

- When `0` is specified for the interval, HiRDB does not use a time interval for collecting synchronization point dumps.
- If no transactions execute during an interval, no synchronization point dump is collected even though the amount of the time specified here has elapsed.

Specification guidelines

- This operand need not be specified if no specific amount of time is specified for restarting HiRDB.
- The value specified for this operand affects the amount of time required to restart HiRDB.
- Specifying a small value for this operand reduces the amount of time required for database recovery during a HiRDB restart. However, because the frequency of synchronization point dumps increases, online performance may deteriorate in some cases.

  Conversely, specifying a large value for this operand increases the amount of time required for database recovery during a HiRDB restart. However, because the frequency of synchronization point dumps decreases, online performance may improve in some cases.

Tuning the specified value

The synchronization point dump collection interval can be checked with the statistics analysis utility; the relevant information is shown under `SYNC POINT GET INTERVAL` in the statistical information related to system operation. The average `SYNC POINT GET INTERVAL` value should be used. If the synchronization point dump collection interval is determined to be too long, the specification value should be decreased; conversely, if it is determined to be too short, the specification value should be increased.

Note

The synchronization point dump collection interval is determined on the basis of the volume of system log information that is output. Therefore, committing data from memory to a database takes a long time during intervals that have few updating transactions. If an error occurs at such a time, it will take longer to recover the transactions that were generated during that period. If this is a possibility, the synchronization point dump collection interval should be set by also using the `interval` value.

Relationship to other operands

If `v6compatible or v7compatible` is specified in the `pd_sysdef_default_option` operand, the default system log output volume becomes `10000`.

4.3.14 Operands related to server status files (when an error occurs)

For details about the measures to be taken when an error occurs in a status file, see the *HiRDB Version 8 System Operation Guide*.

58) pd_sts_initial_error = stop | continue | excontinue

When a server (single server, front-end server, dictionary server, or back-end server) starts, HiRDB performs a process of identifying the current server status file. This operand specifies the action that HiRDB takes when any of the following errors is detected during this identification process.

- No real server status file is found.
- An error is detected in the server status file.

The current file identification process is applied to the server status files specified by the `pd_sts_file_name_1-7` operands.

If Real Time SAN Replication based on the log-only synchronous method is used, the status files for log application processing for the server specified in the `pd_sts_subfile_name_1` through `pd_sts_subfile_name_7` operands are also used at the log application site. For the log application site, replace the `pd_sts_last_active_file` and `pd_sts_last_active_side` operands in the above description with the `pd_sts_last_active_subfile` and `pd_sts_last_active_side_sub` operands, respectively.

`stop`:

When an error is detected in the server status file during the current file identification process, the startup of the server is stopped and HiRDB (or a unit for a HiRDB/Parallel Server) is not started. In this case, first take a corrective action for the status file in which the error was detected, and then start HiRDB.

continue or excontinue:

Even when an error is detected in the server status file during the current file identification process, the startup of the server is continued if the current file is normal. However, the startup may be stopped depending on the value specified for the pd_sts_singleoperation operand (whether operation should continue with a single status file). The following table shows the relationship to the pd_sts_singleoperation operand.

- **Relationship to the pd_sts_singleoperation operand**

pd_sts_singleoperation operand value	Processing by HiRDB	HiRDB administrator's action
continue	When an error is detected in the server status file, HiRDB cannot identify the current file, and thus the startup of the server is stopped, and HiRDB (or a unit for a HiRDB/ Parallel Server) is not started.	The HiRDB administrator identifies the current file and specifies the pd_sts_last_active_file and pd_sts_last_active_side operands. Afterwards, start HiRDB.
stop (default value)	When an error is detected in the server status file, HiRDB identifies the current file and the startup of the server is continued. However, if File A or B satisfies any of the conditions listed in the following table (cases in which HiRDB cannot identify the current file), the startup of the server is stopped, and HiRDB (or a unit for a HiRDB/ Parallel Server) is not started.	If HiRDB cannot identify the current file, the HiRDB administrator identifies the current file and specifies the pd_sts_last_active_file and pd_sts_last_active_side operands. Afterwards, start HiRDB.

- **When HiRDB cannot identify the current file**

pd_sts_initial_error operand value	File A status	File B status
continue	Error shutdown	Error shutdown
	Error shutdown	Open (initial state)
	Error shutdown	No real file
	Open (initial state)	Error shutdown
	Open (initial state)	No real file
	No real file	Open (initial state)
	No real file	Error shutdown
	No real file	No real file
excontinue	Error shutdown	Error shutdown
	Error shutdown	No real file
	No real file	Error shutdown
	No real file	No real file

Therefore, to minimize actions by the HiRDB administrator (to reduce the number of cases in which both the `pd_sts_last_active_file` and `pd_sts_last_active_side` operands must be specified), specify the following:

- `pd_sts_initial_error = excontinue`
- `pd_sts_singleoperation = stop`

Specification guidelines

The following table shows the specification guidelines, along with their advantages and disadvantages.

Item	pd_sts_initial_error operand value	
	stop	continue or excontinue
Processing by HiRDB during server startup	When an error is detected in a server status file, the startup of the server is stopped and HiRDB (or a unit for a HiRDB/Parallel Server) is not started.	Even when an error is detected in a server status file, the startup of the server is continued if the current file is normal.

Item	pd_sts_initial_error operand value	
	stop	continue or excontinue
Specification guideline	To improve system reliability, specify `stop`.	To simplify the error-handling actions during HiRDB startup, specify `continue` or `excontinue`.
Advantage	Guarantees that all server status files of the server are normal when the server starts. Therefore, if an error occurs in the current file after HiRDB has started, it can be swapped to a spare file.	Even when an error is detected in some server status file during server startup, HiRDB can start with the remaining normal files only. Therefore, HiRDB stop time can be shortened. In this case, because the number of spare files has become small, it is necessary to immediately repair the status files containing errors.
Disadvantage	Possibility increases that an error in a server status file stops the startup of HiRDB.	Because HiRDB may be running with only a small number of spare files, system reliability is low. Depending on the number of spare files available, it may not be possible to swap server status files.

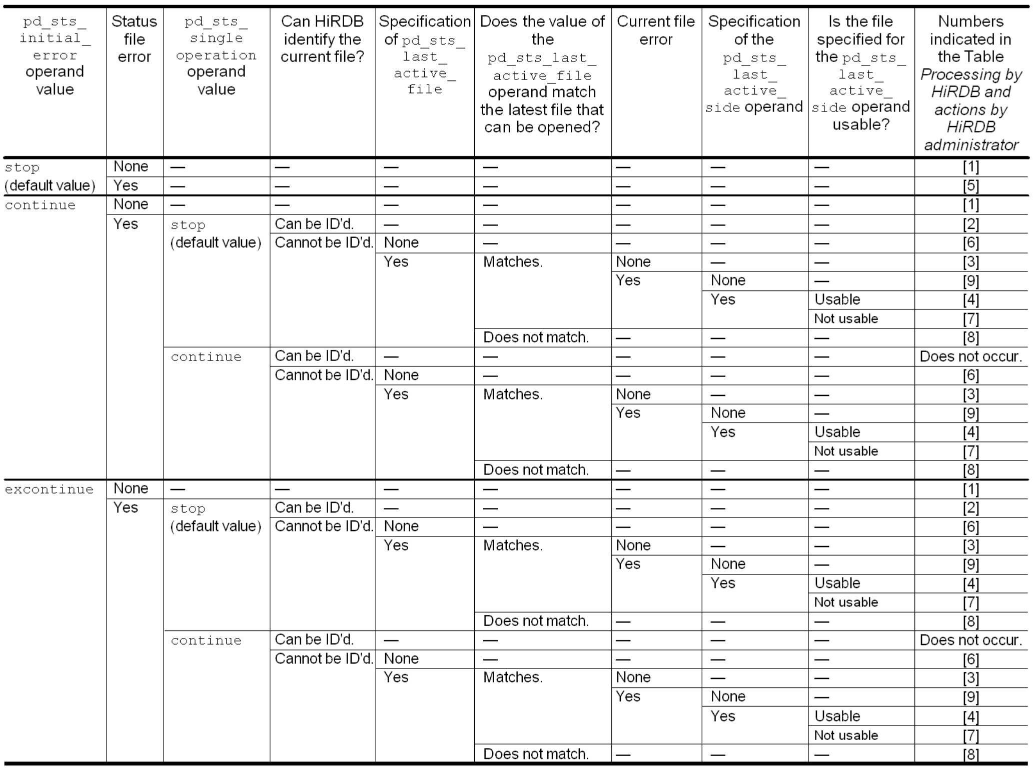
Notes

- If both current files are abnormal, the startup of the server is stopped regardless of the value specified for this operand, and HiRDB (or a unit for a HiRDB/Parallel Server) is not started.
- Before starting HiRDB, do not initialize the current file with the `pdstsinit` command. If the current file is initialized, HiRDB cannot be restarted.

Remarks

The following figure shows specification values for this operand, processing by HiRDB, and actions to be taken by the HiRDB administrator.

- Specification values for this operand and processing by HiRDB



`pd_sts_initial_error` operand value	Status file error	`pd_sts_single operation` operand value	Can HiRDB identify the current file?	Specification of `pd_sts_last_active_file`	Does the value of the `pd_sts_last_active_file` operand match the latest file that can be opened?	Current file error	Specification of the `pd_sts_last_active_side` operand	Is the file specified for the `pd_sts_last_active_side` operand usable?	Numbers indicated in the Table *Processing by HiRDB and actions by HiRDB administrator*
`stop` (default value)	None	—	—	—	—	—	—	—	[1]
	Yes	—	—	—	—	—	—	—	[5]
`continue`	None	—	—	—	—	—	—	—	[1]
	Yes	`stop` (default value)	Can be ID'd.	—	—	—	—	—	[2]
			Cannot be ID'd.	None	—	—	—	—	[6]
				Yes	Matches.	None	—	—	[3]
						Yes	None	—	[9]
							Yes	Usable	[4]
								Not usable	[7]
					Does not match.	—	—	—	[8]
		`continue`	Can be ID'd.	—	—	—	—	—	Does not occur.
			Cannot be ID'd.	None	—	—	—	—	[6]
				Yes	Matches.	None	—	—	[3]
						Yes	None	—	[9]
							Yes	Usable	[4]
								Not usable	[7]
					Does not match.	—	—	—	[8]
`excontinue`	None	—	—	—	—	—	—	—	[1]
	Yes	`stop` (default value)	Can be ID'd.	—	—	—	—	—	[2]
			Cannot be ID'd.	None	—	—	—	—	[6]
				Yes	Matches.	None	—	—	[3]
						Yes	None	—	[9]
							Yes	Usable	[4]
								Not usable	[7]
					Does not match.	—	—	—	[8]
		`continue`	Can be ID'd.	—	—	—	—	—	Does not occur.
			Cannot be ID'd.	None	—	—	—	—	[6]
				Yes	Matches.	None	—	—	[3]
						Yes	None	—	[9]
							Yes	Usable	[4]
								Not usable	[7]
					Does not match.	—	—	—	[8]

—: Not applicable. (The condition does not affect the processing by HiRDB.)

- Processing by HiRDB and actions by HiRDB administrator

Corresponding No.	Processing by HiRDB	HiRDB administrator's action
[1]	HiRDB startup processing continues.	None
[2]	HiRDB identifies the latest current file and continues the startup processing.	Make the file that is in the error-shutdown state into a spare file.
[3]	Using the file specified in the `pd_sts_last_active_file` operand as the current status file, HiRDB startup processing continues.	Make the file that is in the error-shutdown state into a spare file.

Corresponding No.	Processing by HiRDB	HiRDB administrator's action
[4]	Using the files specified in the `pd_sts_last_active_file` and `pd_sts_last_active_side` operands as the current status files, HiRDB startup processing continues.	Make the file that is in the error-shutdown state into a spare file.
[5]	Because `stop` is specified for the `pd_sts_initial_error` operand, HiRDB startup processing is stopped.	See the manual *HiRDB Version 8 Messages*, and take the corrective action indicated by the reason code `0000000010` in message `KFPS01005-E`.
[6]	Because the current file that was being used during the previous operation cannot be identified, HiRDB startup processing is stopped.	See the manual *HiRDB Version 8 Messages*, and take the corrective action indicated by the reason code `0000000016` in message `KFPS01005-E`.
[7]	Because the normal current file identified by HiRDB does not match the file specified in the `pd_sts_last_active_side` operand, HiRDB startup processing is stopped.	See the manual *HiRDB Version 8 Messages*, and take the corrective action indicated by the reason code `0000000017` in message `KFPS01005-E`.
[8]	Because the current file name identified by HiRDB does not match the file name specified in the `pd_sts_last_active_file` operand, HiRDB startup processing is stopped.	See the manual *HiRDB Version 8 Messages*, and take the corrective action indicated by the reason code `0000000015` in message `KFPS01005-E`.
[9]	Because the normal current file that was being used during the previous operation cannot be identified, HiRDB startup processing is stopped.	See the manual *HiRDB Version 8 Messages*, and take the corrective action indicated by the reason code `0000000018` in message `KFPS01005-E`.

59) pd_sts_singleoperation = stop | continue

Specifies whether or not processing of server status files may continue in the single operation mode. The status file single-operation mode means that processing is to continue using only the normal file (single file) when an error occurs in the status file and a spare file is not available. For details about the status file single-operation mode, see the *HiRDB Version 8 Installation and Design Guide*.

If an error occurs in one of the current files and a spare file is available, the status file is swapped and processing continues, regardless of the value specified for this operand (operation in the single-operation mode does not occur).

If Real Time SAN Replication based on the log-only synchronous method is used, the status files for log application processing for the server are also used at the log application site.

`stop`:

Does not permit operation in the single-operation mode. If operation in the single-operation mode is necessary, HiRDB (or a unit for a HiRDB/Parallel Server) is abnormally terminated. If HiRDB is abnormally terminated, allocate spare files and then restart HiRDB.

`continue`:

Enables operation in the single-operation mode. When the single-operation mode goes into effect, the message `KFPS01044-I` is output. If an error occurs in the normal file during operation in the single-operation mode, or if HiRDB (or a unit for a HiRDB/Parallel Server) is abnormally terminated while the status file is being updated, HiRDB cannot be restarted. Therefore, when the single-operation mode goes into effect, immediately allocate spare files.

Specification guidelines

- Hitachi recommends that you specify `stop` to increase system reliability. Hitachi also recommends that you increase the number of spare files to guard against errors in the current files.
- The following table shows the specification guidelines, along with their advantages and disadvantages.

Item	**pd_sts_singleoperation operand value**	
	stop	**continue**
Specification guideline	To improve system reliability, specify `stop`.	Specify `continue` if it is important not to stop HiRDB.
Advantage	When an error occurs in one of the current files and a spare file is not available, operation in the single-operation mode does not occur and HiRDB is abnormally terminated. Consequently, the possibility of losing the content of the current files is reduced.	Even when an error occurs in one of the current files and a spare file is not available, processing can be continued. Therefore, the possibility that an error in the status file stops HiRDB is reduced.
Disadvantage	Possibility that an error in the status file stops HiRDB increases. However, increasing the number of spare files can reduce this possibility.	If an error occurs in the normal status file during operation in the single-operation mode or if HiRDB is abnormally terminated during updating of the status file, the content of the current file is lost and HiRDB cannot be restarted.

Relationship to other operands

The combination of the values specified for the `pd_sts_singleoperation` and `pd_sts_initial_error` operands determines the processing by HiRDB when an error occurs in the status file. Therefore, the values to be specified for these operands should be determined together.

4.3.15 Operands related to the BES connection holding facility

For details about the BES connection holding facility, see the *HiRDB Version 8 System Operation Guide*.

60) pd_bes_connection_hold = Y | N

This operand is applicable only to a HiRDB/Parallel Server.

Specifies whether to use the BES connection holding facility.

`Y`: The BES connection holding facility is used.

`N`: The BES connection holding facility is not used.

Relationship to the client environment definition

The value of this operand can be changed for each client. To change the operand for a client, specify the `PDBESCONHOLD` operand in the client environment definition. For details about the `PDBESCONHOLD` operand, see the *HiRDB Version 8 UAP Development Guide*.

Note

When you use the BES connection holding facility, make sure that the following condition is satisfied:

number-of-processes-in-each-back-end-server (value of the `pd_max_bes_process` operand) ≥ *number-of-all-front-end-server-processes* (value of the `pd_max_users` operand × *number-of-front-end-servers*)

If this condition is not satisfied, a shortage in the number of back-end server processes may cause an SQL error. Furthermore, if you plan to execute a program such as a utility while HiRDB is running, additionally, allocate the number of back-end server processes required by the utility.

61) pd_bes_conn_hold_trn_interval = *back-end-server-connection-hold-time*

~ <unsigned integer>((0-3600))<<1>>(seconds)

Specifies the BES connection holding period in seconds.

When the BES connection holding facility is used, HiRDB monitors the period between the termination of a transaction and the execution of the next transaction.

If this period is within the specified value, the connection between the front-end server and the back-end server is maintained. However, if this period exceeds the specified value, the connection between the front-end server and the back-end server is terminated after the transaction is terminated.

If 0 is specified for this operand, the period is not monitored. The connection between the front-end server and the back-end server is terminated only when the connection between the front-end server and a client is terminated by SQL DISCONNECT (xa_close if the XA library is being used) or because the value of the PDCWAITTIME operand is exceeded.

Chapter

5. Single Server Definition

This chapter explains the operands of the single server definition.

This chapter contains the following sections:

5.1 Operand formats

A single server definition defines information for a single server. This section explains the formats used to specify the operands of a single server definition. Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *5.2 Operands whose specification values can be changed* and *5.3 Operand explanations*.

For users who are creating HiRDB system definitions for the first time

The first step is to determine the values to be specified for the operands shown in boldface type. In principle, HiRDB can be started once the boldface operands have been specified.

No.	Format	Operand category
1	`[set pd_process_count =` *resident-processes-count*`[,` *resident-processes-count-at-server-startup*`]]`*	Processes
2	`[set pd_server_cleanup_interval =` *interval-for-stopping-nonresident-server-processes*`]`*	
3	`[set pd_svr_castoff_size =` *maximum-memory-size-used-by-server-process*`]`*	
4	`[set pd_max_open_fds =` *maximum-number-of-files-and-pipes-accessed-by-process*`]`	
5	`[set pd_max_ard_process =` *asynchronous-READ-process-count*`]`*	
6	`[set pd_dfw_awt_process =` *number-of-processes-to-be-written-in-parallel-in-deferred-write*`]`	
7	`[set pd_work_buff_mode = each |pool]`*	Work tables
8	`[set pd_work_buff_size =` *work-table-buffer-size*`]`*	
9	`[set pd_work_buff_expand_limit =` *work-table-buffer-expansion-limit*`]`*	
10	`[set pd_watch_pc_client_time =` *maximum-client-request-wait-time*`]`*	System monitoring
11	`[set pd_spd_syncpoint_skip_limit =` *maximum-number-of-skipped-synchronization-point-dumps*`]`*	

No.	Format	Operand category
12	[set pd_dfw_syncpoint_skip_limit = *maximum-number-of-skipped-synchronization-point-dumps-resulting-from-delay-of-synchronization-point-dump-acquisition-due-to-deferred-write-processing**	
13	[set pd_cwaittime_wrn_pnt = *output-condition-for-SQL-runtime-warning-information (% specification)* | *output-condition-for-SQL-runtime-warning-information (time specification)*]*	SQL runtime warning output facility
14	[set pd_uap_exerror_log_use = YES | NO]	Facility for output of extended SQL error information
15	[set pd_lck_pool_size = *server-lock-pool-size*]*	Lock
16	[set pd_lck_until_disconnect_cnt = *total-number-of-tables-and-RDAREAs-to-be-locked-per-server-UNTIL-DISCONNECT-specification*]*	
17	[set pd_max_open_holdable_cursors = *maximum-number-of-holdable-cursors-that-can-be-concurrently-open-when-LOCK-statement-with-UNTIL-DISCONNECT-specification-is-not-executed*]	
18	[set pd_lck_hash_entry = *lock-pool-hash-entry-count*]*	
19	[set pd_dbsync_lck_release_count = *global-buffer-lock-release-interval-during-synchronization-point-processing*]*	
20	[set pd_sql_object_cache_size = *SQL-object-buffer-size*]*	Buffers
21	[set pd_table_def_cache_size = *table-definition-information-buffer-size*]*	
22	[set pd_auth_cache_size = *user-privilege-information-buffer-size*]*	
23	[set pd_view_def_cache_size = *view-analysis-information-buffer-size*]*	
24	[set pd_alias_cache_size = *table-alias-definition-information-buffer-size*]*	
25	[set pd_type_def_cache_size = *user-defined-type-information-buffer-size*]*	

No.	Format	Operand category
26	[set pd_routine_def_cache_size = *routine-definition-information-buffer-size*]*	
27	[set pd_registry_cache_size = *registry-information-buffer-size*]*	
28	[set pd_sds_shmpool_size = *single-server-shared-memory-size*]*	Shared memory
29	[set pd_rpc_trace = Y | N]*	RPC trace information
30	[set pd_rpc_trace_name = "*name-for-RPC-trace-collection-files*"]*	
31	[set pd_rpc_trace_size = *RPC-trace-collection-file-size*]*	
32	[set pd_module_trace_max = *maximum-number-of-module-traces-that-can-be-stored*]*	Troubleshooting information
33	[set pd_module_trace_timer_level = *module-trace-output-time-acquisition-method*]	
34	[set pd_max_add_dbbuff_no = *maximum-global-buffers-count-for-dynamic-addition*]*	Global buffer
35	[set pd_max_add_dbbuff_shm_no = *maximum-shared-memory-segments-count-for-dynamic-addition*]*	
36	[set pd_audit_def_buffer_size = *security-audit-information-buffer-length*]*	Security audit facility
37	[set pd_node_name = *local-node-RD-node-name*]	Distributed database
38	[set pd_plugin_ixmk_dir = "*index-information-file-creation-directory-name*" *or* "*index-information-file-creation-HiRDB-file-system-area-name*"]*	Delayed batch creation of plug-in index
39	[set pd_java_stdout_file = "*Java-virtual-machine-standard-output-or-standard-error-output-destination-file*"]*	Java
40	**[set pd_log_dual = Y | N]***	System log files

No.	Format	Operand category
41	[set pd_log_dual_write_method = serial | parallel]	
42	[set pd_log_remain_space_check = warn | safe]*	
43	[set pd_log_auto_unload_path = "*unload-log-file-output-directory*" *[,"unload-log-file-output-directory*"]...]	
44	[set pd_log_singleoperation = Y | N]*	
45	[set pd_log_rerun_reserved_file_open = Y | N]*	
46	[set pd_log_rerun_swap = Y | N]*	
47	[set pd_log_swap_timeout = *wait-time-for-completion-of-system-log-file-swapping*]*	
48	[set pd_log_unload_check = Y | N]*	
49	[set pd_log_max_data_size = *log-input/output-buffer-size*]*	
50	[set pd_log_write_buff_count = *log-output-buffer-sectors-count*]*	
51	[set pd_log_rec_leng = *system-log-file-record-length*]*	
52	**[set pd_spd_dual = Y | N]***	Synchronization point dump files
53	[set pd_spd_assurance_msg = Y | N]*	
54	[set pd_spd_assurance_count = *number-of-guaranteed-valid-generations*]*	
55	[set pd_spd_reduced_mode = *reduced-mode-operation-option*]*	
56	[set pd_spd_reserved_file_auto_open = Y | N]*	

No.	Format	Operand category
57	[set pd_spd_max_data_size = *synchronization-point-dump-file-buffer-size*]*	
58	[set pd_log_sdinterval = *system-log-output-volume*[,*interval*]]*	
59	**set pd_sts_file_name_1 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"**	Server status files
	[set pd_sts_file_name_2 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_3 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_4 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_5 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_6 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_7 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
60	[set pd_sts_subfile_name_1 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing* ","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_2 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing* ","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_3 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing* ","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_4 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing* ","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_5 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing* ","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_6 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing* ","*secondary-status-file-name-for-log-application-processing*"]	

No.	Format	Operand category
	[set pd_sts_subfile_name_7 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
61	[set pd_sts_initial_error = stop | continue | excontinue]*	Server status files (when an error occurs)
62	[set pd_sts_singleoperation = stop | continue]*	
63	[set pd_sts_last_active_file = "*logical-file-name*"]	
64	[set pd_sts_last_active_side = A | B]	
65	[set pd_sts_last_active_subfile = "*logical-file-name*"]	
66	[set pd_sts_last_active_side_sub = A | B]	
67	**pdwork -v "*HiRDB-file-system-area-name*"[,"*HiRDB-file-system-area-name*"]...**	Work table files
68	**{{pdlogadfg -d sys -g *file-group-name* [ONL]}}**	System log file configuration
69	**{{pdlogadpf -d sys -g *file-group-name* -a "*system-log-file-name*" [-b "*system-log-file-name*"]}}**	
70	**{{pdlogadfg -d spd -g *file-group-name* [ONL]}}**	Synchronization point dump file configuration
71	**{{pdlogadpf -d spd -g *file-group-name* -a "*synchronization-point-dump-file-name*" [-b "*synchronization-point-dump-file-name*"]}}**	
72	{{set pdlogadfg -d ssp -g *file-group-name* [ONL]}}	
73	{{set pdlogadpf -d ssp -g *file-group-name* -a "*synchronization-point-dump-file-name-for-log-application-processing*" [-b "*synchronization-point-dump-file-name-for-log-application-processing*"]}}	
74	{{[pdplgprm -n *plug-in-name* [-s *shared-memory-size*]]}}	Plug-ins

* If this operand is omitted, the value specified in the same operand in the server common definition is used. However, for the following operands, the value specified for the same operand in the system common definition, rather than the server common definition, is used:

- pd_cwaittime_wrn_pnt

5.2 Operands whose specification values can be changed

The values of some of the single server definition operands can be changed in the system common definition, unit control information definition, and server common definition. These operands are indicated below. After a planned termination, forced termination, or abnormal termination of HiRDB, some HiRDB system definition operands can be modified while others cannot be modified. The operands that can be modified are indicated below.

No.	Operand	Definition name				Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	SDS		
1	pd_process_count	—	—	Y	Y	Y[2]	Y[2]
2	pd_server_cleanup_interval	—	—	Y	Y	Y	Y
3	pd_svr_castoff_size	—	—	Y	Y	Y	Y
4	pd_max_open_fds	—	—	—	Y	Y	Y
5	pd_max_ard_process	—	—	Y	Y	Y	Y
6	pd_dfw_awt_process	—	—	Y	Y	Y	Y
7	pd_work_buff_mode	—	—	Y	Y	Y	Y
8	pd_work_buff_size	—	—	Y	Y	Y	Y
9	pd_work_buff_expand_limit	—	—	Y	Y	Y	Y
10	pd_watch_pc_client_time[4]	—	—	Y	Y	Y	Y
11	pd_spd_syncpoint_skip_limit	—	—	Y	Y	Y	Y
12	pd_dfw_syncpoint_skip_limit	—	—	Y	Y	Y	Y

No.	Operand	Definition name				Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	SDS		
13	pd_cwaittime_wrn_pnt[4]	Y	—	—	Y	Y	Y
14	pd_uap_exerror_log_use[4]	—	—	—	Y	Y	Y
15	pd_lck_pool_size	—	—	Y	Y	Y[2]	Y
16	pd_lck_until_disconnect_cnt	—	—	Y	Y	N	Y
17	pd_max_open_holdable_cursors	—	—	Y	Y	N	N
18	pd_lck_hash_entry	—	—	Y	Y	Y	Y
19	pd_dbsync_lck_release_count	—	—	Y	Y	Y	Y
20	pd_sql_object_cache_size	Y	—	Y	Y	N	Y
21	pd_table_def_cache_size	—	—	Y	Y	Y[2]	Y[2,3]
22	pd_auth_cache_size	—	—	Y	Y	Y[2]	Y[2,3]
23	pd_view_def_cache_size	—	—	Y	Y	Y[2]	Y[2,3]
24	pd_alias_cache_size	—	—	Y	Y	Y[2]	Y[2,3]
25	pd_type_def_cache_size	—	—	Y	Y	Y	Y
26	pd_routine_def_cache_size	—	—	Y	Y	Y	Y
27	pd_registry_cache_size	—	—	Y	Y	Y	Y
28	pd_sds_shmpool_size	—	—	Y	Y	Y[2]	Y[2]

No.	Operand	Definition name				Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	SDS		
29	pd_rpc_trace	Y	Y	Y	Y	Y	Y
30	pd_rpc_trace_name	Y	Y	Y	Y	Y	Y
31	pd_rpc_trace_size	Y	Y	Y	Y	Y	Y
32	pd_module_trace_max	Y	Y	Y	Y	Y	Y
33	pd_module_trace_timer_level	Y	Y	Y	Y	Y	Y
34	pd_max_add_dbbuff_no	—	—	Y	Y	N	Y
35	pd_max_add_dbbuff_shm_no	—	—	Y	Y	N	Y
36	pd_audit_def_buffer_size	—	—	Y	Y	Y	Y
37	pd_node_name	—	—	—	Y	Y	Y
38	pd_plugin_ixmk_dir	—	—	—	Y	N	Y
39	pd_java_stdout_file	Y	Y	Y	Y	Y	Y
40	pd_log_dual	—	—	Y	Y	N	N
41	pd_log_dual_write_method	—	—	Y	Y	Y	Y
42	pd_log_remain_space_check	—	—	Y	Y	Y	Y
43	pd_log_auto_unload_path	—	—	—	Y	N	N
44	pd_log_singleoperation	—	—	Y	Y	N	N

No.	Operand	Definition name				Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	SDS		
45	`pd_log_rerun_reserved_file_open`	—	—	Y	Y	Y	Y
46	`pd_log_rerun_swap`	—	—	Y	Y	Y	Y
47	`pd_log_swap_timeout`	—	—	Y	Y	Y	Y
48	`pd_log_unload_check`	—	—	Y	Y	Y	Y
49	`pd_log_max_data_size`	—	—	Y	Y	Y[2,3]	Y[2]
50	`pd_log_write_buff_count`	—	—	Y	Y	Y[3]	Y
51	`pd_log_rec_leng`	—	—	Y	Y	Y	Y
52	`pd_spd_dual`	—	—	Y	Y	N	N
53	`pd_spd_assurance_msg`	—	—	Y	Y	Y	Y
54	`pd_spd_assurance_count`	—	—	Y	Y	N	N
55	`pd_spd_reduced_mode`	—	—	Y	Y	N	Y
56	`pd_spd_reserved_file_auto_open`	—	—	Y	Y	N	Y
57	`pd_spd_max_data_size`	—	—	Y	Y	N	N
58	`pd_log_sdinterval`	—	—	Y	Y	Y	Y
59	`pd_sts_file_name_1-7`	—	—	—	Y	Y[1]	Y[1]
60	`pd_sts_subfile_name_1-7`	—	—	—	Y	N	N

No.	Operand	Definition name				Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	SDS		
61	`pd_sts_initial_error`	—	—	Y	Y	Y	Y
62	`pd_sts_singleoperation`	—	—	Y	Y	Y	Y
63	`pd_sts_last_active_file`	—	—	—	Y	Y	Y
64	`pd_sts_last_active_side`	—	—	—	Y	Y	Y
65	`pd_sts_last_active_subfile`	—	—	—	Y	Y	Y
66	`pd_sts_last_active_side_sub`	—	—	—	Y	Y	Y
67	`pdwork`	—	—	—	Y	N	N
68	`pdlogadfg -d sys`	—	—	—	Y	Y[1]	Y[1]
69	`pdlogadpf -d sys`	—	—	—	Y	Y[1]	Y[1]
70	`pdlogadfg -d spd`	—	—	—	Y	Y[1]	Y[1]
71	`pdlogadpf -d spd`	—	—	—	Y	Y[1]	Y[1]
72	`pdlogadfg -d ssp`	—	—	—	Y	Y[1]	Y[1]
73	`pdlogadpf -d ssp`	—	—	—	Y	Y[1]	Y[1]
74	`pdplgprm`	—	—	—	Y	Y	Y

Y: Yes, specification value can be modified.

N: No, specification value cannot be modified.

—: Specification value cannot be modified because the operand is not applicable.

SYS: System common definition

UNT: Unit control information definition

SVR: Server common definition

SDS: Single server definition

[1] Operand that can be added only; it cannot be deleted or modified.

[2] If the specified value is too small, it may not be possible to restart HiRDB. When this is the case, restore the value to what is was before the change and then restart HiRDB.

[3] If the specified value is too large, it may not be possible to restart HiRDB. When this is the case, restore the value to what is was before the change and then restart HiRDB.

[4] The following values for the HiRDB system definition operands can be different for each client. To change a value for a client, specify the operand in the client environment definition. For details about the client environment definition, see the *HiRDB Version 8 UAP Development Guide*.

HiRDB system definition operands	Client environment definition operands
pd_watch_pc_client_time	PDSWATCHTIME
pd_cwaittime_wrn_pnt	PDCWAITTIMEWRNPNT
pd_uap_exerror_log_use	PDUAPEXERLOGUSE

5.3 Operand explanations

5.3.1 Operands related to processes

1) pd_process_count = *resident-processes-count*[,*resident-processes-count-at-server-startup*]

～ <unsigned integer>((0-2000))

resident-processes-count

Specifies the number of processes that can be made resident in the single server. A resident process is a process that is activated at the time the server is started.

Advantage

By activating the processes used by transactions that can be processed concurrently by the single server at the time of system startup and keeping them resident, the process startup time can be reduced even when new transactions are entered. However, HiRDB startup will take longer.

Specification guidelines

See *Specification guidelines* for the `pd_process_count` operand of the server common definition.

Tuning the specified value

For details about how to tune the resident processes count, see the *HiRDB Version 8 System Operation Guide*.

Notes

- Because the resident processes count has an effect on memory space availability, specifying an unnecessarily large number may prevent HiRDB from starting.
- If more processes than the resident process count are needed, additional processes are dynamically started, up to the maximum processes count allowed. However, depending on the value specified for the `pd_max_server_process` operand, it may not be possible to start all of the processes indicated by the maximum processes count.

Operand rule

See *Operand rules* for the `pd_process_count` operand of the server common definition.

resident-processes-count-at-server-startup

Specifies the number of processes that can be made resident during HiRDB startup.

It is common for resident processes to be activated during HiRDB startup. If there are many resident processes, the amount of time required for HiRDB startup increases proportionately. For example, it takes approximately 1 second to activate a process on a server machine with a 100-MIPS performance rating.

Following are the differences in processing that result depending on whether or not a resident processes count at server startup is specified:

- When there is no specification of a resident processes count at server startup (when, for example, `pd_process_count = 500` is specified)

  All 500 resident processes are activated during HiRDB startup, and HiRDB will not start until they are all activated. In this example, it would take a 100-MIPS server machine about 500 seconds to activate all the resident processes during HiRDB startup.

- When a resident processes count at server startup is specified (when, for example, `pd_process_count = 500, 50` is specified)

  Some of the resident processes (50 in this case) are activated during HiRDB startup, and the remaining resident processes are activated after HiRDB startup. HiRDB can be started as long as the specified number of resident processes are activated. In this example, it would take a 100-MIPS server machine about 50 seconds to activate 50 resident processes during HiRDB startup. The remaining 450 resident processes (in this example) would be activated after HiRDB startup.

Advantage

The amount of time required for HiRDB startup is reduced. You should use this option when you want to reduce the HiRDB startup time as much as possible, such as when you are using the system switchover facility.

Specification guideline

The specified value should be equal to the number of processes that will be required immediately after HiRDB startup is completed.

Notes

When you specify a resident processes count at server startup, you should recheck the value in the `PDCWAITTIME` operand of the client environment definition.

If more UAPs than the resident processes count at server startup are to be executed immediately following HiRDB startup, transaction processing may not be performed until after the remaining resident processes have been

activated. Therefore, if the value specified in the PDCWAITTIME operand of the client environment definition is small, it may not be possible to process some UAPs due to timeouts. For details about the PDCWAITTIME operand, see the *HiRDB Version 8 UAP Development Guide*.

2) pd_server_cleanup_interval = *interval-for-stopping-nonresident-server-processes*

~ <unsigned integer>((0-1440)) (minutes)

Specifies in minutes the interval for checking for nonresident server processes in HiRDB that are to be stopped. The facility for stopping nonresident server processes is applied when the number of executing server processes exceeds the number of processes that can be made resident (value specified by the pd_process_count operand). The number of server processes that the facility stops is computed automatically by HiRDB.

Advantages

This facility improves the utilization rates of the memory and of process resources because it increases the number of nonresident processes that can be reused when the workload (number of server processes that are executing) peaks.

Specification guidelines

- For example, if there is only one hour a day during which peak workload occurs and the intervals between peaks within that hour are approximately two minutes apart, 2 should be specified for this operand.
- This facility does not have any effect if the number of server processes that execute concurrently during peak periods is always fewer than the number of resident processes. In this case, this operand should be omitted.

Tuning the specified value

Collect statistical information on system operations for each server for one week. Determine the workload peaks from the number of server processes being serviced (# OF PROCESSES ON SERVICE). If that peak value exceeds the currently set number of processes that can be made resident (value specified by the pd_process_count operand), determine the interval between individual peaks and set that number of minutes.

However, if there are ample resources, such as memory and CPU, in the server machine, adding the shortfall in the number of processes to the number of resident processes (i.e., increasing the value of the pd_process_count operand) would be more effective in improving performance than specifying the pd_server_cleanup_interval

operand.

Note

When this operand is omitted or `0` is specified, the system checks every 10 seconds for nonresident server processes that are waiting to be serviced and stops any such processes that are found.

3) pd_svr_castoff_size = *maximum-memory-size-used-by-server-process*

~ <unsigned integer>((0-2048)) (MB)

This operand need not be specified for the Linux version.

Specifies the maximum size of memory used by each server process processed by a single server. If at the applicable trigger point shown below the amount of memory being used by a server process exceeds the value specified here, that server process is terminated. When a server process is so terminated, the `KFPS00350-w` message is output. This is called the *facility for monitoring the memory size of server processes*; for details, see the *HiRDB Version 8 System Operation Guide*.

Server type	Process name	Process termination trigger
Single server	pdsds	At UAP disconnection

Advantages

The facility for monitoring the memory size of server processes resolves the following problem:

- *The memory size of a server-resident process becomes too large during a particular SQL processing, significantly reducing the amount of system memory that is available.*

The HiRDB server releases memory space that is no longer needed. However, even when a program releases memory, the OS holds the memory area itself in the memory management facility inside the applicable process. Consequently, a process that becomes large in terms of using a large memory area even once never shrinks and continues to have an adverse impact on the system, especially in the case of resident processes. The facility for monitoring the memory size of server processes can prevent memory shortages because it terminates even resident processes.

Application criterion

Apply this facility when the amount of memory space used by a HiRDB server process becomes large, resulting in memory shortages.

Specification guidelines

- Maximum value for this operand

  Normally specify a value by considering the maximum processing capability of HiRDB. Assuming that the maximum number of SQL statements may be concurrently connected, determine for this operand a value that satisfies the following condition in each unit:

  $a \times (b + c) < d$

  a: Number of server processes within the unit (*maximum-number-of-concurrent-connections* × *server-count-inside-unit*)

  b: Virtual memory size for one server process immediately following HiRDB startup

  c: `pd_svr_castoff_size` operand value

  d: Memory size that can be allocated to a unit (memory size excluding the area used by other programs)

- Minimum value for this operand

  If the value specified for this operand is smaller than the memory size needed for normal SQL processing, the efficiency of making processes resident will deteriorate, resulting in frequent process terminations and restarts. Each time this occurs, a message is output to the syslogfile or the message log file, resulting in further performance degradation. To prevent this, select for this operand a value that satisfies the following condition for each server:

  $a - b < c$

  a: Server process virtual memory size following SQL process termination

  b: Virtual memory size for server process immediately following HiRDB startup

  c: `pd_svr_castoff_size` operand value

The virtual memory size can be determined by an OS command (e.g., the `top` command in HP-UX).

Operand rule

When `0` is specified for this operand, the facility for monitoring the memory size of server processes is not applied.

4) pd_max_open_fds = *maximum-number-of-files-and-pipes-accessed-by-process*

～ <unsigned integer>((1-8192))<<320>>

Specifies the maximum number of files and pipes accessed by HiRDB processes.

Specification guidelines

- Omit this operand if the maximum number of the plug-in index storage RDAREAs updated in a single transaction is less than 100.
- If the maximum number of the plug-in index storage RDAREAs updated in a single transaction is equal to or greater than 100, specify the maximum number + 320 for this operand.

Notes

The maximum value of this operand differs depending on the OS type as follows:

- HP-UX: 8192
- Solaris: 2048
- 64-bit mode Solaris: 8192
- AIX: 8192
- Linux: 1024

If an index information file is created during utility execution, files and pipes are accessed up to the smaller of the following two values regardless of the value specified for this operand:

- The physical upper limit for the number of files that can be opened or locked by a single process of the OS's operating system parameter
- The maximum value of this operand for each OS

The physical upper limit for the number of files that can be opened or locked by a single process of the OS's operating system parameter is described below.

- For HP-UX: Value of `maxfiles_lim`
- For Solaris: Value of `rlim_fd_max`
- For AIX: Value of `nofiles_hard`
- For Linux: Value of `NR_OPEN`

5) pd_max_ard_process = *asynchronous-READ-process-count*

～ <unsigned integer>((0-256))

Specify this operand if you use the asynchronous READ facility. For this operand, specify the number of processes necessary for asynchronous READ operations. For details on the asynchronous READ facility, see the *HiRDB Version 8 Installation and Design Guide*.

Condition

A value of `1` or greater must be specified for the `-m` option of the `pdbuffer` operand.

Advantage

The asynchronous READ facility is especially effective (improves performance) when character special files, for which input and output take a long time, are used. Conversely, when regular files, for which input and output do not take a long time, or SANRISE disks are used, the asynchronous READ facility may not have much effect because of the following reasons:

- Input/out time does not overlap with CPU time most of the time.
- Communication processing has a large overhead.

Specification guidelines

- Specify `0` or `1`. However, if a value between 2 and 256 is specified for the `-m` option of the `pdbuffer` operand, specify the same value as the `-m` option value. If a value greater than 256 is specified for the `-m` option, specify the same value as the number of disk devices that store RDAREAs and system files or `256`.
- Increasing the value of this operand can shorten the processing time when the degree of concurrency is high for the SQL statements to which the asynchronous READ facility is applied. Decreasing the value of this operand may lengthen the processing time when the degree of concurrency is high for the SQL statements to which the asynchronous READ facility is applied. This is because asynchronous READ processes may have to wait for processing completion.
- Because a number of processes equaling *value-of-this-operand* × *server-count* are started, determine a value for this operand by taking resources (shared memory and message queue) into consideration. For the method of estimating shared memory and message queue sizes, see the *HiRDB Version 8 Installation and Design Guide*.

Tuning the specified value

For the method of tuning the specification value (the number of asynchronous READ processes), see the *HiRDB Version 8 System Operation Guide*.

Relationship to other operands

If you change the value of this operand, re-evaluate the value of the `pd_max_server_process` operand.

Operand rule

If you specify 0 for this operand, the asynchronous READ facility is not used.

6) pd_dfw_awt_process = ***number-of-processes-to-be-written-in-parallel-in-deferred-write***

~ <unsigned integer>((2-255))

Specify this operand when you use the *facility for parallel writes in deferred write processing* for all buffer pools. Specify for this operand the number of processes to be processed in parallel. Increasing the number of processes can shorten the write processing time. For details about the facility for parallel writes in deferred write processing, see the *HiRDB Version 8 Installation and Design Guide*.

Specification guidelines

Specify 2, which is the smallest value that enables the facility for parallel writes in deferred write processing. Furthermore, to determine the value for this operand, see *Tuning deferred write processing* in the manual *HiRDB Version 8 Installation and Design Guide*.

Note

Specifying the facility for parallel writes in deferred write processing increases the number of processes and consequently raises the CPU usage rate.

5.3.2 Operands related to work tables

7) pd_work_buff_mode = each | pool

Specifies the method of allocating buffers when HiRDB creates tables.

each: Allocate a buffer for each work table.

pool: Allocate a buffer pool for each server process.

Specification guidelines

- Normally, pool is specified. pool is the appropriate specification when a large volume of data is to be retrieved and when manipulations such as join, ORDER BY, and GROUP BY are to be performed.
- When the size of the process private area that can be used for work table buffers is predetermined, pool should be specified. When pool is specified, HiRDB efficiently allocates work table buffers to work tables.

  In such a case, the process private area is occupied on the basis of the value specified in pd_work_buff_size, and input/output operations on work tables are buffered in that pool. Therefore, the process private memory is occupied only to the extent of the value specified in pd_work_buff_size.

8) pd_work_buff_size = *work-table-buffer-size*

~ <unsigned integer> (KB)

- 32-bit mode: **((128-1000000))**
- 64-bit mode: **((128-4000000000))**

Specifies in kilobytes the size of buffers for work tables to be created by HiRDB.

Item	pd_work_buff_mode=each is specified	pd_work_buff_mode=pool is specified
Advantage	A large work table buffer size reduces the number of I/O operations associated with data manipulation, which means that the execution time of SQLs that use work tables is also reduced. However, because each server's process private memory is used, specification of this option should take into account the overall size of the system memory (real memory and virtual memory). If pd_work_buff_mode = each is specified, the memory size to be allocated is *value of* pd_work_buff_size × *required number of work tables*. A process private memory that has been allocated will remain allocated to that process at least in the virtual memory (will not be released until that process disappears) even after release is requested by HiRDB. Therefore, specifying an unnecessarily large value may cause a virtual memory shortage for other processes.	
Application criterion	pd_work_buff_mode = pool should be specified when a large volume of data is to be retrieved and when manipulations such as join, ORDER BY, and GROUP BY are to be performed.	

Item	pd_work_buff_mode=each is specified	pd_work_buff_mode=pool is specified
Specification guidelines	• Specify the size of the buffer to be allocated for one work table. • If a value greater than the work table memory capacity is specified for the work table buffer size, input/output to the work table file during work table creation is eliminated, thus reducing the time necessary for work tables. The following formula can be used to determine the work table memory capacity: `Work table memory capacity = Applicable work table size`* ÷ `2`	• Specify the size of the buffer pool to be allocated for the entire server process. • Specify a value between 4352 and 5120 when a large volume of data is to be retrieved or when manipulations such as join, `ORDER BY`, and `GROUP BY` are to be performed. Specifying such a value increases the unit of sorting input/output, thus reducing the sort time. • If a large value is specified for the work table buffer size, with the total work table memory capacity for each SQL statement as the upper limit, input/output operations on the work table file during work table creation is eliminated, thus reducing the time necessary for work tables. The following formula can be used to determine the total work table memory capacity for each SQL statement: `Total work table memory capacity per SQL statement =` $a \times b + c \times d$
Notes	When multiple users execute processes concurrently or when an SQL statement that uses multiple work tables is executed, the specified size buffer is allocated for each work table. Consequently, specifying a large value may result in a memory shortage.	If the value specified for the work table buffer size is smaller than the number of work tables to be used by each SQL statement, the processing time may become longer than when `each` is specified. Specifically, specify a value that is at least equal to *maximum number of work tables for each SQL statement* × *128*. The following formula can be used to determine the maximum number of work tables for each SQL statement: `Maximum number of work tables for each SQL statement =` $b + d$
Operand rule	Specify a multiple of 128. If the value is not a multiple of 128, it is rounded up to the next multiple of 128.	• Specify a multiple of 128. If the value is not a multiple of 128, it is rounded up to the next multiple of 128. • Specify at least 384. If a value that is smaller than 384 is specified, it is rounded up to 384.

a:

↑ {Capacity of work table (for storing column information)* (KB) ÷ 2} ÷

128↑ × 128

b:

Maximum number of work tables (for storing column information)*

c:

↑{Capacity of work table (for storing positional information)* (KB) ÷ 2} ÷ 128↑ × 128

d:

Maximum number of work tables (for storing positional information)*

* For details about how to determine these values, see the *HiRDB Version 8 Installation and Design Guide*.

9) pd_work_buff_expand_limit = ***work-table-buffer-expansion-limit***

~ <unsigned integer> (KB)

- 32-bit mode: **((128-1000000))**
- 64-bit mode: **((128-4000000000))**

The size of the work table buffer to be created by HiRDB is specified by the `pd_work_buff_size` operand. Specify the `pd_work_buff_expand_limit` operand if you want to automatically expand a work table buffer when the space in this buffer becomes insufficient. The work table buffer is expanded up to the size specified by this operand.

For example, when the following values are specified for the operands, a 1,024-KB work table buffer is normally allocated. When this size becomes insufficient, the work table buffer is expanded up to 2,048 KB.

- `pd_work_buff_size = 1024`
- `pd_work_buff_expand_limit = 2048`

HiRDB expands a work table buffer in the following cases:

- The necessary work table buffer cannot be allocated when hash execution is applied to an execution method that uses hash join or subquery hash as the joining method.
- A 128-KB work table buffer allocated to each work table becomes insufficient when multiple work tables are concurrently used.

Condition

The `pd_work_buff_mode` operand must be omitted, or `pool` must be specified for it.

Advantage

You can prevent a work table buffer shortage (too small a value specified for the `pd_work_buff_size` operand) from causing UAP errors.

Notes

- A work table buffer is not expanded when either of the following conditions is satisfied:
  - The `pd_work_buff_expand_limit` operand is not specified.
  - `pd_work_buff_expand_limit` operand value ≤ `pd_work_buff_size` operand value
- When a work table buffer is expanded, memory is freed. To do this, resident single server processes are terminated when UAPs are disconnected from HiRDB (except in the Linux version).

Operand rule

Specify a multiple of 128. If a value other than a multiple of 128 is specified, it is automatically rounded up to a multiple of 128.

Relationship to other operands

When a work table buffer is expanded for the first time in a single server process, the `KFPH29008-I` message is output. Note that you can use the `pd_work_table_option` operand to suppress this message output.

Note

After a work table buffer has been expanded, when the number of work tables being used by the applicable server process goes to zero, the expanded work table buffer is released. The number of work tables being used can go to zero in the following cases:

- All cursors that were being used are closed. (In this case, the number of work tables being used may not go to zero.)
- A transaction is normally terminated or cancelled when a holdable cursor is not being used.
- A UAP is disconnected from HiRDB when a holdable cursor is being used.

Remarks

Hash join, subquery hash execution is applied in the following cases:

- *Application of optimizing mode 2 based on cost* and *hash join, subquery hash execution* are specified in the `pd_additional_optimize_level` operand, the

PDADDITIONALOPTLVL operand of the client environment definition, or the ADD OPTIMIZE LEVEL operand of the SQL compile option.

- HASH is specified for the SQL optimization specification of the joining method inside an SQL statement.
- HASH is specified for the SQL optimization specification of the subquery execution method inside an SQL statement.

5.3.3 Operands related to system monitoring

10) pd_watch_pc_client_time = *maximum-client-request-wait-time*

~ <unsigned integer>((0-65535)) (Seconds)

Specifies in seconds the maximum amount of time for a server to wait for the next request from a HiRDB client after the HiRDB server returns a response to a request from a Windows-compatible HiRDB client.

If no request comes from the HiRDB client within the specified amount of time, it will be assumed that an error occurred at the client and the connection between the server and the client will be terminated forcibly. No notice of disconnection is sent to the HiRDB client in such a case.

The time that is monitored is the period between CONNECT and DISCONNECT (i.e., the non-transaction status time), excluding the period between SQL execution startup and COMMIT or ROLLBACK.

Notes

- When 0 is specified, the server waits indefinitely for the next request from the HiRDB client.
- If a small value (e.g., up to 600) is specified for this operand, the HiRDB client may detect *server down* during SQL execution and may be terminated incorrectly.
- For a UNIX version of a HiRDB client (including a Linux version of a HiRDB client), time is not monitored, regardless of the value specified for this operand. To monitor time for a UNIX version of a HiRDB client, specify the PDSWATCHTIME client environment definition of the HiRDB client.

Relationship to client environment definition

The value of this operand can be modified for each client. To do so, the PDSWATCHTIME operand must be specified in the client environment definition. For details about the PDSWATCHTIME operand, see the *HiRDB Version 8 UAP Development Guide*.

11) pd_spd_syncpoint_skip_limit =

maximum-number-of-skipped-synchronization-point-dumps

~ <unsigned integer>((0, 2-100000))

If an event such as an endless loop occurs in a UAP, acquisition of synchronization point dumps may be skipped. If several synchronization point dumps are not acquired (instead, they are skipped), there will be an increase in the number of overwrite-disabled system log files, which may result in a shortage of system log file capacity and ultimately in abnormal termination of the unit.

This operand specifies the maximum number of times synchronization point dumps can be skipped (number of skips per transaction). When the number of skipped synchronization point dumps reaches the specified operand value, the target transaction is cancelled forcibly and rolled back. This is called the *skipped effective synchronization point dump monitoring facility*.

If Real Time SAN Replication based on the log-only synchronous method is used, the skipped effective synchronization point dump monitoring facility cannot be used at the log application site regardless of this operand value.

For details on this facility, see the *HiRDB Version 8 System Operation Guide*.

Advantage

Specifying this operand provides a means of dealing with endless loops in UAPs.

Specification guideline

Normally, specify `0` for this operand. When `0` is specified, HiRDB computes the upper limit for the skip count. If the specification of `0` causes a problem, change the value of this operand. For the method of estimating the specification value, see the *HiRDB Version 8 System Operation Guide*.

Notes

- The monitoring provided for by the specification of this operand begins the first time a synchronization point dump is collected after HiRDB is started (including a restart).
- When this operand is specified, UAPs that are being executed in the no-log mode are also monitored. If the processing of a UAP being executed in the no-log mode is cancelled, the database cannot be automatically recovered, and thus RDAREAs are placed in the error shutdown state. Therefore, when specifying the upper limit, account also for the size of the system logs that are output from other transactions inside the server during the processing of UAPs that are executed in the no-log mode.
- The `pdload`, `pdmod`, `pdrorg`, `pdexp`, `pddbst`, `pdgetcst`, `pdrbal`, and `pdvrup` commands are not monitored by this facility.

12) pd_dfw_syncpoint_skip_limit = *maximum-number-of-skipped-synchronization-point-dumps-resulting-from-delay-of-synchronization-point-dump-acquisition-due-to-deferred-write-processing*

～ <unsigned integer>((0-100000))

If a synchronization point occurs within the synchronization point dump acquisition interval but before deferred write processing is completed, acquisition of the synchronization point dump is skipped. This is because acquisition of the synchronization point dump is delayed by the deferred write processing and the number of update buffers output by the synchronization point exceeds the number of update buffers that can be output within the synchronization point dump acquisition interval.

If more than one synchronization point dump is skipped, there will be an increase in the number of system log files that cannot be overwritten, which may result in a shortage of system log file capacity and ultimately in abnormal termination of the unit.

This operand specifies the maximum number of times synchronization point dumps can be skipped (number of skips per transaction) due to deferred write processing.

When the number of skipped synchronization point dumps due to deferred write processing reaches the specified operand value, HiRDB determines the maximum number of update buffers in such a manner that acquisition of a synchronization point dump can be completed within the synchronization point dump acquisition interval. If the maximum number of update buffers is exceeded, HiRDB then outputs the oldest update buffer and limits the total size of update buffers at a synchronization point. This is called the *update buffer size restriction facility*.

Advantage

If a synchronization point occurs within the synchronization point dump acquisition interval but before deferred write processing has terminated, the unit terminates abnormally. This operand enables such abnormal termination of the unit to be avoided.

Specification guidelines

Normally, you will omit this operand. If you wish to prevent unit abnormal termination caused by the occurrence of a synchronization point before termination of deferred write processing within the synchronization point dump acquisition interval, specify 1.

If an acceptable number of times synchronization point dumps can be skipped can be determined in advance, such as from the size of the log information, specify that value.

Operand rules

If 0 is specified in this operand, HiRDB does not use the update buffer size restriction facility.

Notes

The following notes explain the period of effectiveness of the update buffer size restriction facility. The period of effectiveness means the interval between issuance of the `KFPH23035-I` message and issuance of the `KFPH23036-I` message.

- If the number of update buffers exceeds the maximum number of update buffers determined by HiRDB, update buffers are output after update processing has executed; this degrades the update processing throughput. You can use the following formula to obtain the maximum number of update buffers determined by HiRDB:

(synchronization point dump interval / unit value of `WRITE`[#])
x (1 - (amount of log information from the previous synchronization point dump to the pre-synchronization point)
x (number of buffer sectors in buffer pool / total number of buffer sectors in buffer pool that was updated at the synchronization point)

  #: For details about the unit value of `WRITE`, see the *HiRDB Version 8 System Operation Guide*.

- If the deferred write trigger is specified in the `pd_dbbuff_rate_updpage` or `pdbuffer -y` operand, and each operand value becomes greater than the maximum number of update buffers determined by HiRDB, the maximum number of update buffers determined by HiRDB is changed to the number of update buffers that triggers deferred write processing.

  The value of the `pdbuffer -w` operand is adjusted automatically so that up to the maximum number of update buffers is output for each buffer.

- A skipped synchronization point dump is detected during update buffer output processing at a synchronization point. Therefore, the update buffer size restriction facility may be enabled after a synchronization point dump is skipped and an error message is displayed.

- Normally, there is one output request per synchronization point for the number of processes that used the facility for parallel writes in deferred write processing during synchronization point processing (`pd_dfw_awt_process`).

  However, if the update buffer size restriction facility is used, there will be more than one output request in order to facilitate early detection of skipped synchronization point dumps.

5.3.4 Operands related to SQL runtime warning output facility

13) pd_cwaittime_wrn_pnt =
output-condition-for-SQL-runtime-warning-information (% specification) **|**
output-condition-for-SQL-runtime-warning-information (time specification)

Specify this operand when using the SQL runtime warning output facility. You can specify this operand using one of the following two methods:

- Specifying a percentage
- Specifying a time duration

For details on the SQL runtime warning output facility, see the *HiRDB Version 8 System Operation Guide*.

output-condition-for-SQL-runtime-warning-information (% specification)**:**
~ <unsigned integer>((0-99)) or <unsigned decimal number>((0-99.999999)) (%)

Specifies the condition for outputting SQL runtime warning information of the SQL runtime warning output facility as a percentage of the maximum client wait time (value of the `PDCWAITTIME` operand in the client environment definition). After SQL execution, HiRDB checks the SQL execution time. If this SQL execution time is determined to exceed the time specified by this operand, the following warning information is output. This is called the *SQL runtime warning output facility*.

- SQL runtime warning information file
- Warning message (`KFPA20009-W`)

Operand specification methods

Specify this operand as a percentage (%) of the value of the `PDCWAITTIME` operand. For example, if `100` (seconds) is specified for the `PDCWAITTIME` operand and `90` (%) is specified for the `pd_cwaittime_wrn_pnt` operand, HiRDB checks the SQL execution time after SQL execution. If the determined SQL execution time is 90 seconds or longer but less than 100 seconds, the warning information is output.

Example

```
PDCWAITTIME = 100
pd_cwaittime_wrn_pnt = 90
```

output-condition-for-SQL-runtime-warning-information (time specification)**:**
~ <unsigned decimal number>((0-65534.999999))sec (seconds)

Specifies the condition for outputting SQL runtime warning information of the SQL runtime warning output facility as a time duration. After SQL execution, HiRDB checks the SQL execution time. If this SQL execution time is determined to exceed the time specified by this operand, the following warning information is output. This is called the *SQL runtime warning output facility*.

- SQL runtime warning information file
- Warning message (`KFPA20009-W`)

Operand specification method

Specify the time duration (in seconds) to be used as the output trigger. (You can specify up to the sixth decimal.) Add `sec` to the specified value.

Example

```
pd_cwaittime_wrn_pnt = 0.001sec
```

The following explanation is the same for both the percentage and time duration specification.

Operand rule

If `0` or `0sec` is specified in this operand, no warning information is output. (The SQL runtime warning output facility is not used.)

Relationship to client environment definition

You can change the value of this operand for each client. To change it for each client, specify the `PDCWAITTIMEWRNPNT` operand of the client environment definition. For details on the `PDCWAITTIMEWRNPNT` operand, see the *HiRDB Version 8 UAP Development Guide*.

Relationship to other operands

This operand is related to the following operands:

- `pd_cwaittime_report_dir`
- `pd_cwaittime_report_size`

5.3.5 Operand related to the facility for output of extended SQL error information

14) pd_uap_exerror_log_use = YES | NO

Specifies whether to use the facility for output of extended SQL error information. For details about this facility, see the *HiRDB Version 8 UAP Development Guide*.

`YES`:

Uses the facility for output of extended SQL error information. SQL error information is output in an error log file and an SQL error report file.

NO:

The facility for output of extended SQL error information is not used.

Relationship to client environment definition

The value of this operand can be changed for each client. To change the operand for a client, specify the PDUAPEXERLOGUSE operand in the client environment definition. If both this operand and the PDUAPEXERLOGUSE operand in the client environment definition are specified, the PDUAPEXERLOGUSE operand takes precedence.

For details about the PDUAPEXERLOGUSE operand, see the *HiRDB Version 8 UAP Development Guide*.

5.3.6 Operands related to lock

15) pd_lck_pool_size = ***server-lock-pool-size***

~ **<unsigned integer> (KB)**

- 32-bit mode: **((1-2000000))**
- 64-bit mode: **((1-2000000000))**

Specifies in kilobytes the size of the shared memory to be used by the single server for locking.

Specification guidelines

- In the 32-bit mode, the amount of lock pool space required for six lock requests is 1 KB.
- In the 64-bit mode, the amount of lock pool space required for four lock requests is 1 KB.
- The following formulas can be used to determine the value to be specified for this operand:

HiRDB type	Formula
HiRDB/Single Server (32-bit mode)	$\uparrow a \div 6 \uparrow$ (KB)
HiRDB/Single Server (64-bit mode)	$\uparrow a \div 4 \uparrow$ (KB)

a:

Total number of transaction lock requests to be executed concurrently by the single server. The number of lock requests depends on the SQL. For details about how to determine the total number of lock requests, see *E. Determining the Number of Locked Resources*.

Tuning the specified value

See the usage rate for the locked resources management table (`% OF USE LOCK TABLE`) displayed in the statistical information on system operation by the statistics analysis utility. If the maximum usage rate equals or exceeds 80%, it is recommended that the operand's value be increased in preparation for future database expansion. If the maximum usage rate does not exceed 10%, it is recommended that the operand's value be decreased to conserve shared memory space.

Note

If the value specified for this operand is too small, an SQL may return an error.

16) pd_lck_until_disconnect_cnt = *total-number-of-tables-and-RDAREAs-to-be-locked-until-disconnect-specification*

~ <unsigned integer>((0-140000))

Specifies the number of tables and RDAREAs to be held across transactions and the number of resources to be locked for inner replica configuration management and replica group configuration management. According to the value specified for this operand, tables, RDAREAs, and the blocks for which lock with `UNTIL DISCONNECT` is specified in inner replica configuration management and replica group configuration management are allocated in the shared memory.

Specification guidelines

Normally, this operand need not be specified. Specification of a value other than the default value may be necessary in the following cases:

- When the number of utilities to be executed concurrently increases
- When a holdable cursor is used
- When the local buffer specified in the `pdlbuffer` operand is used

For the method of estimating the specification value for this operand, see *D.4 Formula for determining total number of tables and RDAREAs per server locked with UNTIL DISCONNECT specified (pd_lck_until_disconnect_cnt)*.

Tuning the specified value

If the value specified for this operand is small, a transaction may roll back or a utility may terminate abnormally with return code `8`. In such cases, the message `KFPA11914-E` or `KFPH28001-E` is output. If this occurs, the value of this operand should be increased.

When the value of this operand is increased, the amount of required memory space increases proportionately. The required memory size can be expressed as follows: *value-of-this-operand* × 48 (64 in the 64-bit mode) bytes.

17) pd_max_open_holdable_cursors = ***maximum-number-of-holdable-cursors-that-can-be-concurrently-open-when-LOCK-statement-with-UNTIL-DISCONNECT-specification-is-not-executed***

~ <unsigned integer>((16-1024))

When you use holdable cursors for a table for which a `LOCK` statement with `UNTIL DISCONNECT` specification is not executed, this operand specifies the maximum number of holdable cursors that can be concurrently open for each transaction.

Note

Specifying a value other than the default value for this operand increases the amount of shared memory used.

Relationship to other operands

The values specified for this operand and the following operands are used for computing the shared memory size for lock servers. For a 32-bit mode HiRDB system, if the values specified for these operands are too large, the shared memory size of the lock servers exceeds 2 GB, and as a result, HiRDB may not start. Therefore, adjust the values specified for these operands so that the shared memory size of the lock servers does not exceed 2 GB.

- `pd_max_access_tables`
- `pd_max_users`
- `pd_lck_hash_entry`
- `pd_lck_pool_size`

For details about shared memory, see the *HiRDB Version 8 Installation and Design Guide*.

18) pd_lck_hash_entry = ***lock-pool-hash-entry-count***

~ <unsigned integer>((0-2147483647))

Specifies the number of hash table entries to be used in the lock pool. According to the value specified here, HiRDB allocates a lock pool in the shared memory for the unit controller.

Specification guidelines

1. Normally, omit this operand.
2. Consider specifying a value for this operand in the following cases:

- If you do not wish to change the shared memory size if possible when upgrading to Version 06-02 or a newer version, specify `11261`. In this case, the same number of hash entries is allocated as in the older version, and the hash table size inside the lock pool remains the same as before.
- Specifying for this operand a value greater than the recommended value may improve performance. However, specifying a value greater than variable *a* described in *Determining the recommended value* does not improve performance over a case in which *a* is specified.

Operand rules

- If `0` is specified for this operand or if this operand and the `pd_lck_hash_entry` operand of the server common definition are omitted, HiRDB computes a recommended value for each server. For details about recommended values, see *Determining the recommended value* for the `pd_lck_hash_entry` operand in the server common definition.
- If a non-zero value or a non-prime number is specified for this operand, HiRDB assumes that the largest prime number not exceeding the specified value has been specified.

Note

If the value specified for this operand is too small, hash entry shortage may occur, resulting in performance degradation. If this operand is omitted, neither hash entry shortage nor performance degradation due to hash entry shortage occurs.

Determining the recommended value

For details on the recommended values, see *Determining the recommended value* for the `pd_lck_hash_entry` operand of the server common definition.

19) pd_dbsync_lck_release_count = ***global-buffer-lock-release-interval-during-synchronization-point-processing***

~ <unsigned integer>((0, 100-1073741824))

Specifies an interval for unlocking global buffers, when global buffer locking occurs during synchronization point processing.

During synchronization point processing, search processing occurs on the buffers (update buffers) and must be applied to the disk. Normally, global buffers are unlocked at a specific interval during search processing on the update buffers.

For example, if 100 is specified in this operand, a global buffer is unlocked once when search processing on 100 sectors (global buffer sectors) is completed. After that, the global buffer is locked again and search processing is resumed. In this example, unlocking occurs once every 100 sectors.

Advantage

By specifying this operand, you can adjust the global buffer lock time during synchronization point processing. When a small value is specified in this operand, the global buffer lock time becomes short and transaction performance may improve during synchronization point processing.

To obtain the global buffer pool lock time, execute the statistics analysis utility and in the global buffer pool statistical information check the item called `Buffer pool lock time during synchronization point processing (SYNCL).`

Specification guidelines

Normally, there is no need to specify this operand. Consider specifying this operand when both the following conditions apply:

- Transaction performance drops during synchronization point processing.
- A large number of buffer sectors is specified in the `-n` option of the `pdbuffer` operand.

Operand rules

- If the specified value is in the range 1 to 99, 100 is set automatically.
- If 0 is specified, global buffers are locked until update buffer search processing is completed.

Notes

If a small value is specified in this operand, the update buffer search time becomes longer due to interruptions by other transactions. The global buffers updated during that time are also output during synchronization point processing. Therefore, the number of update buffers to be output during synchronization point processing increases. To obtain the number of update buffers to be output during synchronization point processing, execute the statistics analysis utility and in the global buffer pool statistical information check the item called `Number of synchronization point output pages (SYNCW).`

5.3.7 Operands related to buffers

20) pd_sql_object_cache_size = *SQL-object-buffer-size*

～ <unsigned integer> (KB)

- 32-bit mode: **((22-256000))**

- 64-bit mode: **((22-2000000))**

Specifies in kilobytes the size of the buffer area (shared memory) in which SQL objects are to be placed.

Specification guidelines

- SQL objects are saved in a buffer until the user's transaction has terminated. The buffer must be large enough to store the SQL objects of all transactions that will be executed concurrently.
- SQL analysis can be reduced by saving the SQL objects of static SQLs in the buffer after transaction termination (until the buffer runs out of space) and sharing them among multiple users who execute the same UAP. To effectively utilize the buffer, it should be allocated so that the SQL objects of frequently-used UAPs are resident in the buffer.
- To estimate the buffer size, first determine the buffer size needed for UAP execution from the length of the SQL objects from the SQL statements to be issued by the UAP. Then, compute the buffer size by considering the number of UAPs that will be executed concurrently and the number of concurrently executing users.
- For details about how to estimate the length of the SQL object from a single SQL statement, see *D.2 Formulas for determining size of SQL object buffer (pd_sql_object_cache_size).*

Tuning the specified value

For details about how to tune the SQL object buffer size, see the *HiRDB Version 8 System Operation Guide*.

21) pd_table_def_cache_size = ***table-definition-information-buffer-size***

~ <unsigned integer> (KB)

- 32-bit mode: **((100-65535))**
- 64-bit mode: **((100-2000000))**

Specifies in kilobytes the size of the buffer (shared memory) for table definition information that has been used. Table definition information is used during pre-processing of an SQL statement.

Advantages

- Table definition information that has been used is kept in the memory as long as possible and thus can be used subsequently without an input operation.
- Performance improves when a large number of dynamic SQLs are used.

Specification guidelines

- The total size of the definition information for frequently-used tables is specified.
- For details about how to determine the size of the table definition information buffer per table, see *D.3 Formulas for determining size of table definition information buffer (pd_table_def_cache_size)*.

Tuning the specified value

For details about how to tune the size of the table definition information buffer, see the *HiRDB Version 8 System Operation Guide*.

22) pd_auth_cache_size = *user-privilege-information-buffer-size*

~ <unsigned integer>((1-100)) (KB)

Specifies in kilobytes the size of the buffer (shared memory) for user privilege information.

Specification guidelines

- The user privilege information buffer stores CONNECT privilege, DBA privilege, and audit privilege information. If this buffer contains no information, information is obtained from a dictionary table during HiRDB connection, thus lengthening the response time. Therefore, specify a buffer size that can store the information for the users who are always connected.
- Storing the user privilege information of each user requires 68 bytes. Use this information when computing the total buffer size.

Tuning the specified value

For details about how to tune the size of the user privilege information buffer, see the *HiRDB Version 8 System Operation Guide*.

23) pd_view_def_cache_size = *view-analysis-information-buffer-size*

~ <unsigned integer> (KB)

- 32-bit mode: **((0-32000))**
- 64-bit mode: **((0-2000000))**

Specifies in kilobytes the size of the buffer (shared memory) for view analysis information.

Advantage

View analysis information that has been used is kept in the shared memory and can be used subsequently without an I/O operation.

Specification guideline

The total size of the view analysis information for frequently-used view tables is specified.

Tuning the specified value

For details about how to tune the size of the buffer for view analysis information, see the *HiRDB Version 8 System Operation Guide*.

24) pd_alias_cache_size = ***table-alias-definition-information-buffer-size***

~ <unsigned integer>((0-3000)) (KB)

Specifies in kilobytes the size of the buffer (shared memory) for table alias definition information.

Specification guidelines

This area should be large enough to hold the information for all frequently used table aliases. The size of the information on each table alias is approximately 164 bytes.

25) pd_type_def_cache_size = ***user-defined-type-information-buffer-size***

~ <unsigned integer> (KB)

- 32-bit mode: **((100-65536))**
- 64-bit mode: **((100-2000000))**

This operand is applicable to user-defined types; specification of this operand is recommended when user-defined types will be used.

Specifies in kilobytes the size of the buffer (shared memory) for information on user-defined types. When a user-defined type is used, information on it is stored in this buffer. This information is used during pre-processing of SQL statements.

Advantages

- When user-defined type information is stored in this buffer, it is not necessary to access the dictionary table when the same user-defined type is used subsequently, thus reducing the number of input operations and the CPU usage time.
- Specification of this operand improves performance when many dynamic SQLs are used.

Specification guidelines

The specified size should be the total of the sizes of the definition information for frequently-used user-defined types that are defined in tables. The following formula can be used to determine the definition information size for one user-defined type:

$\uparrow\{((0.3 + 0.2 \times a + 0.1 \times b) + 3) \div 4\}\uparrow \times 4$ (KB)

a: Number of user-defined type attributes

b: Number of subtypes that have inherited a supertype

Tuning the specified value

For details about how to tune the size of the buffer for user-defined type information, see the *HiRDB Version 8 System Operation Guide*.

26) pd_routine_def_cache_size = *routine-definition-information-buffer-size*

~ <unsigned integer> (KB)

- 32-bit mode: **((0, 20-65536))**
- 64-bit mode: **((0, 20-2000000))**

Specifies in kilobytes the size of the buffer (shared memory) for storing the following types of definition information (this information is used during pre-processing of SQL statements):

- Plug-in facility definition information
- System definition scalar facility definition information
- Routine definition information

Advantages

When these types of definition information are stored in this buffer, it is not necessary to access the dictionary table when the same information is used subsequently, thus reducing the number of I/O operations and CPU usage time.

Application criteria

This operand should be specified when a large number of the following types of SQL statements are used:

- SQL statements that use a plug-in
- SQL statements that use the system definition scalar facility
- SQL statements that use a routine

Specification guidelines

For details about how to determine the value for this operand, see *D.5 Formulas for determining size of routine definition information buffer (pd_routine_def_cache_size)*.

Tuning the specified value

For details about how to tune the size of the routine definition information buffer, see the *HiRDB Version 8 System Operation Guide*.

Note

If the value specified for this operand is smaller than the total of the sizes of the definition information for all plug-ins, the definition information on plug-in facilities will not be allocated in the buffer.

27) pd_registry_cache_size = *registry-information-buffer-size*

～ <unsigned integer>((0-65536)) (KB)

This operand is related to plug-ins. If you use a plug-in that uses registry information, Hitachi recommends the use of this operand. Hitachi also recommends the use of this operand if you use HiRDB Text Search Plug-in.

Specify the size of the buffer (shared memory) for storing registry information (units: KB). When registry information is used, it is stored in the buffer. Registry information is used during the execution of an SQL statement.

Advantages

- Once registry information is stored in this buffer, it is not necessary to access the registry when the same information is used subsequently, thus reducing the number of I/O operations and CPU usage time.
- Specifying this operand can improve performance when a plug-in that makes frequent use of registry information is used.

Specification guidelines

Use the following formula to determine the registry information buffer size:

$\uparrow(0.3 + a)\uparrow \times b$ (KB)

a: Average registry key length (KB)

The average registry key length can be determined with the following SQL statement:

```
SELECT AVG(KEY_LENGTH) FROM MASTER.SQL_REGISTRY
```

Because the result of this SQL statement is output in bytes, convert it to kilobytes.

b: Number of registry keys registered

The number of registry keys registered can be determined with the following SQL:

```
SELECT COUNT(*)FROM MASTER.SQL_REGISTRY
```

Tuning the specified value

For details about how to tune the size of the buffer for registry information, see the *HiRDB Version 8 System Operation Guide*.

5.3.8 Operands related to shared memory

28) pd_sds_shmpool_size = *single-server-shared-memory-size*

~ **<unsigned integer> (KB)**

- 32-bit mode: **((1-200000))**
- 64-bit mode: **((1-4000000000))**

Specifies the size of the area (units: KB) to be used by a single server as part of the shared memory for the unit controller.

Specification guidelines

Normally, omit this operand. HiRDB computes a value for this operand if the `pd_sds_shmpool_size` operand of the server common definition and single server definition is omitted. HiRDB computes this value based on *Formula for computing the shared memory used by a single server* in the manual *HiRDB Version 8 Installation and Design Guide*. If you change the value of the operand in the explanation of the variables in this formula, HiRDB automatically re-computes the value for the `pd_sds_shmpool_size` operand. Note that the value of the `pd_assurance_index_no` operand is assumed for the *total-number-of-indexes-inside-server* variable inside the formula.

In the formula, for the variables *Number of global buffer pools for index* and *Total number of global buffers (number of pdbuffer operands)*, 500 is assumed in the 32-bit mode and 1000 is assumed in the 64-bit mode.

Tuning the specified value

If any of the following messages is output, increase the specification value of this operand.

- `KFPA20003-E`
- `KFPD00005-E`
- `KFPD00012-E`
- `KFPD00021-E`
- `KFPH20003-E`

Notes

- If an unnecessarily large value is specified for this operand, too large a shared memory area will be allocated. If the value specified for this operand is too small, the following may occur:
  - HiRDB will not start
  - A UAP or utility will not execute
- If you omit this operand, you must specify appropriate values for the `pd_assurance_table_no` and `pd_assurance_index_no` operands.

5.3.9 Operands related to RPC trace information

29) pd_rpc_trace = Y | N

Specifies whether or not RPC trace information is to be collected. HiRDB maintenance information is output in the RPC trace information.

Normally, this operand should be omitted.

`Y`: Collect RPC trace information.

`N`: Do not collect RPC trace information.

Note

Specifying `Y` for this operand degrades communication performance.

30) pd_rpc_trace_name = "*name-of-RPC-trace-collection-file*"

~ <pathname of up to 254 characters>

Specifies an `absolute pathname` for the filename for the RPC trace files. Three RPC trace files are created, with 1, 2, and *l* suffixed to the specified file name.

Note

Files with a maximum size of `pd_rpc_trace_size` *value* × 2 are created under the directory specified by this operand. Attention should be paid to the file capacity.

31) pd_rpc_trace_size = *RPC-trace-collection-file-size*

~ <unsigned integer>((1024-2147483648)) (Bytes)

Specifies in bytes the size of the RPC trace files.

The value specified by this operand determines the size of both RPC trace file 1 and RPC trace file 2.

Specification guidelines

An RPC trace collects in a single file in time order the information on the communications among all processes. If the volume of this information exceeds the specified value, trace data output is switched to the other file, which means that older trace information will be overwritten. When this happens, the amount of trace information that is available may be inadequate, making troubleshooting difficult. For this reason, at least 1,000,000 should be specified for this operand.

Note

The value specified by this operand does not apply to RPC trace file *l*, because its size is fixed at 0 bytes.

5.3.10 Operands related to troubleshooting information

32) pd_module_trace_max = ***maximum-number-of-module-traces-that-can-be-stored***

～ <unsigned integer>((126-16383))

A HiRDB process records the history of the executed functions and macros inside the process private memory. This history is called a *module trace*. This operand specifies the number of module trace records. The content of this history is loaded into the `core` file and is output when a process error occurs.

Specification guidelines

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

Process private memory of the following size is allocated to each process:

In the 32-bit mode: 64 + 48 × value of `pd_module_trace_max` *operand* (bytes)

In the 64-bit mode: 64 + 64 × value of `pd_module_trace_max` *operand* (bytes)

33) pd_module_trace_timer_level = 0 | 10 | 20

Specifies how to acquire the time to be output in module traces. The following table explains the meaning of the value specified for this operand.

Specified value	Time acquisition method
`0`	Time is output in seconds at every module trace output location.
`10`	Time is output in microseconds only at performance-critical module trace output locations, such as those before and after input/output processing, and time is output in seconds at other locations.

Specified value	Time acquisition method
20	Time is output in microseconds at every module trace output location.

Specification guidelines

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

If you specify a value other than 0 for this operand, a function for acquiring time in microseconds is issued, and as a result, system performance may decline.

5.3.11 Operands related to global buffers

34) pd_max_add_dbbuff_no = *maximum-global-buffers-count-for-dynamic-addition*

~ <unsigned integer>((1-32752))

In order to change global buffers dynamically, this operand specifies the maximum number of global buffers (per server) that can be added dynamically by the pdbufmod command.

Condition

Y must be specified in the pd_dbbuff_modify operand.

Specification guidelines

- Estimate the number of global buffers to be added dynamically by the pdbufmod command and then specify a sufficient value based on that value.
- Determine the operand's value in such a manner that the following condition is satisfied:

  Value of pd_max_add_dbbuff_no ≤ 2000000 - number of global buffers allocated per server during HiRDB startup

Notes

Do not specify an unnecessarily large value in this operand. If this operand's value is too large, the shared memory used by HiRDB increases, which may result in a shortage of shared memory and an inability of HiRDB to start.

Relationship to other operands

This operand is related to the following operands:

- SHMMAX
- pdbuffer
- pd_max_add_dbbuff_shm_no

35) pd_max_add_dbbuff_shm_no = *maximum-shared-memory-segments-count-for-dynamic-addition*

～ <unsigned integer>((1-32752))

In order to change global buffers dynamically, specifies the maximum number of shared memory segments (per server) that can be allocated when dynamic addition is performed by the pdbufmod command.

Condition

Y must be specified in the pd_dbbuff_modify operand.

Specification guidelines

Estimate the number of global buffers to be added dynamically by the pdbufmod command and then specify an appropriate value.

Notes

- If the following condition is satisfied, the value of the pd_max_add_dbbuff_no operand is assumed in this operand:

  Value of pd_max_add_dbbuff_shm_no < value of pd_max_add_dbbuff_no

  The value of the pd_max_add_dbbuff_no operand is also assumed when the default value satisfies the above condition.

- Do not specify an unnecessarily large value in this operand. If this operand's value is too large, the shared memory used by HiRDB increases, which may result in a shortage of shared memory and an inability of HiRDB to start.
- If the size of a shared memory segment to be added exceeds the SHMMAX operand value, shared memory is divided into multiple segments based on the SHMMAX operand value as the maximum value. Either increase the SHMMAX operand value based on the size of the shared memory segment to be added or increase the pd_max_add_dbbuff_no operand value so that no shortage occurs when the shared memory is segmented.
- If you have changed the value of this operand, re-evaluate the maximum size of a shared memory segment in the OS parameters, the maximum number of shared memory segments in the system, and the maximum

number of shared memory segments per process.

Relationship to other operands

This operand is related to the following operands:

- `SHMMAX`
- `pdbuffer`
- `pd_max_add_dbbuff_no`

5.3.12 Operands related to the security audit facility

36) pd_audit_def_buffer_size = *security-audit-information-buffer-length*

∼ <unsigned integer>((1-2000000)) (kilobytes)

Specifies (in kilobytes) the buffer size (shared memory) used for storing information for the security audit facility.

Specification guidelines

Use the following formula to determine the security audit information buffer length:

↑0.3 + *a* x 0.25↑ (kilobytes)

a: Number of objects for the narrowing condition that was specified in the audit trail of the security audit facility

Notes

If the amount of memory required by this operand's specification cannot be allocated, HiRDB will not start.

5.3.13 Operands related to distributed database

37) pd_node_name = local-node-RD-node-name

∼ <identifier>((1-30 characters))

This operand is applicable only to the HP-UX and AIX 5L versions.

When HiRDB is used as a client, specifies the local node's RD node name so that local access will be recognized when the specified name is specified as the RD node name in an SQL statement.

Specification of this operand is applicable when the distributed database facility is used. Because this operand increases the amount of memory used by single server processes, it should not be specified except when the distributed database facility is used.

5.3.14 Operands related to delayed batch creation of plug-in index

38) pd_plugin_ixmk_dir = "*index-information-file-creation-directory-name*" or "*index-information-file-creation-HiRDB-file-system-area-name*"

～<pathname>

Specifies the name of the directory under which the index information file for delayed batch creation of a plug-in index is to be created. Specify a HiRDB file system area name in order to create the index information file in a HiRDB file system area. An absolute pathname must be used for the directory name or HiRDB file system area name.

For details about delayed batch creation of a plug-in index, see the *HiRDB Version 8 System Operation Guide*.

Notes

- The directory (or HiRDB file system area) specified here must have been created in advance. If a nonexistent directory (or HiRDB file system area) is specified, an error will result during execution of a UAP that specifies delayed batch creation of plug-in indexes (UAP that is executed in an environment in which `PDPLGIXMK = YES` is specified in the client environment definition).
- Once the UAP has executed, the value specified for this operand must not be changed before delayed batch creation of plug-in indexes is performed by the database reorganization utility. If it is changed, a plug-in index delayed batch creation cannot be performed.

5.3.15 Operands related to Java

Java operands are specified when a Java stored procedure or Java stored function is used. For details about Java stored procedures and Java stored functions, see the *HiRDB Version 8 UAP Development Guide.*

Note that some platforms on which HiRDB is run do not support Java stored procedures and Java stored functions. For details, see *Environment supporting Java stored procedures and functions* in the *HiRDB Version 8 System Operation Guide*.

39) pd_java_stdout_file = *"Java-virtual-machine-standard-output-and-standard-error-output-destination-file"*

～<pathname>

Specifies as an absolute pathname the file to which the standard output and standard error output are to be output in a Java virtual machine. If this operand is omitted, the standard output and standard error output of the Java virtual machine are ignored.

Specification guideline

Because the size of the file specified by this operand is extremely large, this operand is not normally specified. It is recommended that this operand be specified during debugging of a Java stored procedure or Java stored function. There is no limit to the size of the file that can be specified by this operand.

Note

If there are simultaneous writing attempts from multiple processes, their output contents cannot be guaranteed.

Operand rules

- Up to 255 characters can be used for the pathname.

5.3.16 Operands related to system log files

40) pd_log_dual = Y | N

Specifies whether or not dual system log files are to be used.

`Y`: Use dual system log files.

`N`: Do not use dual system log files.

Advantages

When dual system log files are used, HiRDB collects the same system log information in both files, which are called File A and File B. If a failure occurs in one of the files while the collected system log file is being loaded, the file can be loaded from the other file, resulting in higher system reliability.

Relationship to other operands

When use of dual system log files is specified, the name of the File B system log file must be specified with the `pdlogadpf` operand.

41) pd_log_dual_write_method = serial | parallel

This operand is applicable only to the AIX 5L version of HiRDB.

This operand specifies whether to use the facility for parallel output of system logs. For details about this facility, see the *HiRDB Version 8 Installation and Design Guide*.

`serial`: The facility for parallel output of system logs is not used.

`parallel`: The facility for parallel output of system logs is used.

When `parallel` is specified for this operand, HiRDB uses the Asynchronous I/O Subsystem to execute parallel output to the system log files.

Conditions

When you specify `parallel` for this operand, the following conditions must be satisfied:

- `pd_log_dual = Y`
- The system log files are located in character special files.

If either of these conditions is not satisfied, the facility for parallel output of system logs is not used regardless of the value specified for this operand.

Additionally, when you specify `parallel` for this operand, the Asynchronous I/O Subsystem must be installed.

42) pd_log_remain_space_check = warn | safe

Specifies the processing to be performed by HiRDB when the available space in the system log file falls below the warning level. This is called the facility for monitoring the free area for the system log file. For details on this facility, see the *HiRDB Version 8 System Operation Guide*.

`warn`:

When the available space in the system log file falls below the warning level, the `KFPS01162-W` message is output.

`safe`:

When the available space in the system log file falls below the warning level, scheduling of new transactions is suppressed, all transactions inside the server are forcibly terminated, and the `KFPS01160-E` message is output.

Specification guideline

Hitachi recommends the specification of `safe` because it can reduce the probability of abnormal termination of units due to system log file space shortage. However, when `safe` is specified, all transactions inside the server are forcibly terminated when a shortage occurs in the available space in the system log file. Therefore, the design of the system log file requires more accuracy. For details on system log file design, see the *HiRDB Version 8 Installation and Design Guide*.

43) pd_log_auto_unload_path = "*unload-log-file-output-directory*"[,"*unload-log-file-output-directory*"]...

~ <pathname>((1-136 characters))

Specifies as absolute pathnames the unload file output directories when the automatic log unloading facility is to be used for the system log. A HiRDB file system area name must be specified to create the unload log file in a HiRDB file system area. *The directories or HiRDB file system areas specified for this operand* *must be created before HiRDB is started.*

For details about the automatic log unloading facility, see the *HiRDB Version 8 System Operation Guide*.

Specification guidelines

It is important to check the available disk space before specifying a directory, so as to ensure that the created unload log file does not cause a disk space shortage.

If an unload log file cannot be created in the specified directory because of a disk space shortage, the automatic log unloading facility stops. If this is a possibility, creation of multiple directories is recommended.

Note, however, that the database recovery operation of selecting the unload files needed for recovery is simplified somewhat when only one directory is used.

Also keep in mind the following when multiple directories are created:

- It is recommended that directories be specified in different partitions to protect against disk errors.
- If the unload log file cannot be created in a single directory because of a full disk or disk error, create an unload log file under a different directory. HiRDB uses the directories specified by this operand in the order of their specification.

Operand rules

- Up to 128 directories can be specified.
- When multiple directories are specified, the same pathname cannot be specified.

Notes

The automatic log unloading facility cannot be used in the following cases:

- `N` is specified in the `pd_log_unload_check` operand.
- The log application site uses Real Time SAN Replication based on the log-only synchronous method (the `KFPS04689-W` message is displayed).

44) pd_log_singleoperation = Y | N

This operand is applicable when dual system log files are used; it need not be specified when dual system log files are not used.

Specifies whether or not the single-operation mode is to be used for the system log files. Even if an error occurs in a system log file, making no dual system log files available, HiRDB can continue processing using the remaining single normal system log file without being abnormally terminated. This is called *single operation of the system log files*.

The mode in which continuation of processing is permitted only with both system log files available (normal operation mode) is called *double operation of the system log files*.

`Y`: Use single operation of the system log files.

`N`: Do not use single operation of the system log files. Both system log files must always be used.

Condition

This operand is valid only when `pd_log_dual = Y` is specified.

45) pd_log_rerun_reserved_file_open = Y | N

Specifies whether or not a system log file is to be opened automatically.

If no system log file that can be overwritten is available during a HiRDB restart, HiRDB opens a reserved file (if one is available), makes it overwritable, and continues processing. This is called *automatic opening of system log file*.

A reserved file is used in the following cases:

- Between the time of a restart and the time when the first synchronization point dump is collected
- When none of the opened file groups can be overwritten

`Y`: Open a system log file automatically (open and use a reserved file).

`N`: Do not open a system log file automatically (do not use a reserved file).

Advantages

When `Y` is specified, the unit can be restarted as long as a reserved file is available, even though no file that can be swapped in is available during the unit restart.

However, if a file that is in unload wait status is available, the unit is stopped; once that file has been unloaded, the unit can be restarted.

46) pd_log_rerun_swap = Y | N

Specifies whether or not the system log files are to be swapped during a HiRDB restart.

`Y`: Swap the system log files.

`N`: Do not swap the system log files.

Advantage

When Y is specified, there can be a physical separation between the system log files used before and after a restart. Therefore, the system log file that was being used before the restart can be reused during server operation.

47) pd_log_swap_timeout = *wait-time-for-completion-of-system-log-file-swapping*

~ <unsigned integer>((1-32580)) (seconds)

Specifies in seconds the wait time during which swapping of system log files must be completed. If a swap of system log files is not completed within the specified amount of time, the unit terminates abnormally.

Specification guidelines

Normally, there is no need to specify this operand. If it takes a long time to swap system log files for a reason such as poor machine performance, you can specify this operand with a value that is greater than the default value. To detect errors or delays quickly during system log file swapping so that the unit can be terminated (such as because of a disk failure), reduce the value of this operand.

48) pd_log_unload_check = Y | N

Specifies whether or not HiRDB is to check the unload status of system log files.

Y:

Check the unload status (normal operation).

N:

Do not check the unload status. A system log file is placed in swappable status, regardless of its unload status, when both of the following conditions are satisfied:

- It is in overwritable status
- It is in extraction completed status (HiRDB Datareplicator)
- Overwriting permitted status for online reorganization (HiRDB Staticizer Option)

In this case, the system log file operation method is to release checking of unload status. For details about this operation method, see the *HiRDB Version 8 System Operation Guide*.

Advantages

Specifying N provides the following advantages:

- Operations are simplified because the system log file unload operation is eliminated.
- It is not necessary to provide files for storing unload files.

Specification guideline

N should be specified if the system log file will not be needed for database recovery (i.e., if recovery from a backup collection point will be sufficient).

Notes

The following points apply when N is specified:

- Database can be recovered only if backups have been made.
- If this option is specified when the system log file is required for database recovery, it will not be possible to recover the database.

49) pd_log_max_data_size = *log-input/output-buffer-size*

~ <unsigned integer>((32000-523000)) (Bytes)

Specifies in bytes the size of the buffer to be used for system log input/output operations.

Specification guideline

Change the specification value according to the following tuning method.

Tuning the specified value

A value (other than the default value) may need to be specified for this operand after the following types of statistics analysis utility statistical information related to system operation have been checked:

- Number of buffer sectors waiting for input/output (`# OF BUFFER FOR WAIT I/O`)

  If the average number of buffer sectors waiting for input/output significantly exceeds 100, increase the value for this operand so that the average approaches 100.
- Number of waits caused by lack of a current buffer (`# OF WAIT THREAD`)

  If the number of waits caused by lack of a current buffer is not 0, increase the value for this operand.

Relationship to other operands

Use this operand and the `pd_log_write_buff_count` operand to determine the log output buffer size.

50) pd_log_write_buff_count = *log-output-buffer-sectors-count*

~ <unsigned integer>((3-65000))

Specifies the number of buffer sectors to be used for system log output.

Tuning the specified value

A value (other than the default value) may need to be specified for this operand after the following types of statistics analysis utility statistical information related to system operation have been checked:

- Number of times the buffer was full (`# OF BUFFER FULL`)
- Number of waits caused by lack of a current buffer (`# OF WAIT THREAD`)

If either of these values is large, a larger value should be specified in order to improve throughput.

Relationship to other operands

Use this operand and the `pd_log_max_data_size` operand to determine the log output buffer size.

51) pd_log_rec_leng = *system-log-file-record-length*

～ <unsigned integer>((1024, 2048, 4096)) (Bytes)

Specifies the record length for the system log files; the specifiable values are 1024, 2048, and 4096.

The record length specified in the `-l` option of the `pdloginit` command should be specified for this operand.

Notes

- If a value that is different from the record length specified by the `-l` option of the `pdloginit` command is specified for this operand, system log files cannot be opened.
- For details about how to modify the system log file record length, see the *HiRDB Version 8 System Operation Guide*.

5.3.17 Operands related to synchronization point dump files

52) pd_spd_dual = Y | N

Specifies whether to use dual synchronization point dump files.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for the server are also used at the log application site.

`Y`: Uses dual synchronization point dump files.

`N`: Does not use dual synchronization point dump files.

Advantage

When dual synchronization point dump files are used, HiRDB collects the same synchronization point dump in both dump files A and B. Even when an error occurs in one of the files when the collected synchronization point dump is being loaded, the synchronization point dump can be loaded from the other file, thus improving system reliability.

Relationship to other operands

To use dual synchronization point dump files, specify the name of synchronization point dump file B in the `pdlogadpf` operand.

53) pd_spd_assurance_msg = Y | N

Specifies whether or not the `KFPS02183-I` message is to be output when a synchronization point dump is completed.

`Y`: Output the message.

`N`: Do not output the message.

54) pd_spd_assurance_count = ***number-of-guaranteed-valid-generations***

~ <unsigned integer>((1-2))

Specifies the range of system log files to be saved during HiRDB operation as a protection against events such as a synchronization point dump file input error during system recovery. What is specified here is a number of synchronization point dump file generations; the specified value is referred to as the *number of guaranteed-valid generations*. The number of past generations of synchronization point dump files specified here are overwrite disabled.

If Real Time SAN Replication based on the log-only synchronous method is used, this operand is ignored, if specified, at the log application site. The system log files used at the log application site are overwritten by the transaction execution site even if they include system log information that has not been applied.

Advantage

When `2` is specified as the number of guaranteed-valid generations and an error occurs in the most recent synchronization point dump file generation, the system can be recovered using the preceding synchronization point dump file generation, resulting in higher reliability.

Specification guideline

- To improve reliability, `2` should be specified for the number of guaranteed valid generations. However, the number of overwrite disabled synchronization point dump files increases (to two).
- If reliability has been improved with the use of dual synchronization point dump files, Hitachi recommends that you omit this operand or specify `1` for it.

Notes

- The minimum number of synchronization point dump files needed is *number-of-guaranteed-valid-generations* + 1.
- Specifying 2 increases the number of overwrite disabled synchronization point dump files. Moreover, the system log files that correspond to the overwrite disabled synchronization point dump files are also overwrite disabled. Consequently, specifying 2 for the number of guaranteed valid generations increases the number of overwrite disabled system log files. As a result, a shortage may occur in the number of system log files that can be swapped in. To prevent this, it may be necessary to reevaluate the system log file capacity.

55) pd_spd_reduced_mode = *reduced-mode-operation-option*

~ <unsigned integer>((0-2))

Specifies whether or not the reduced mode operation for synchronization point dump files is to be used.

Reduced mode operation is a facility for continuing processing if at least two synchronization point dump files can be used, even if an event such as a file error during HiRDB operation or restart reduces the number of synchronization point dump files to or below the number of required files (*number of guaranteed valid generations*[*] + 1).

[*] Value specified for the `pd_spd_assurance_count` operand.

`0`: Do not use the reduced mode operation.

`1`: Use the reduced mode operation.

`2`: Use the reduced mode operation and issue a warning message whenever a synchronization point dump is being acquired during the reduced mode operation.

56) pd_spd_reserved_file_auto_open = Y | N

Specifies whether or not a synchronization point dump file is to be opened automatically. When `Y` is specified and an error in a synchronization point dump file reduces the number of synchronization point dump files to the number of files necessary for operation (*number of guaranteed valid generations*[*] + 1), HiRDB opens a reserved file (if one is available), makes it overwritable, and continues processing. This is called *automatic opening of synchronization point dump file*.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for the server are also used at the log application site.

[*] Value specified for the `pd_spd_assurance_count` operand.

`Y`:

Open a synchronization point dump file automatically. A reserved file is opened when the number of files falls below the number necessary for operation (*number of guaranteed valid generations* + 1).

`N`:

Do not open a synchronization point dump file. No reserved file is opened, even though the number of files falls below the number necessary for operation (*number of guaranteed valid generations* + 1).

Relationship to other operands

This operand has a higher priority than the `pd_spd_reduced_mode` operand.

57) pd_spd_max_data_size = *synchronization-point-dump-file-buffer-size*

~ <unsigned integer>((32000-4000000)) (Bytes)

Specifies in bytes the size of the buffer (shared memory) to be used for synchronization point dump file input/output operations.

The value specified here affects the number of synchronization point dump file input/output operations.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for the server are also used at the log application site.

Specification guidelines

- Normally, this operand need not be specified.
- When the value of this operand is increased, the number of synchronization point dump file input/output operations is reduced.

58) pd_log_sdinterval = *system-log-output-volume*[,*interval*]

Specifies the collection interval for synchronization point dumps. The following should be taken into consideration in specifying this operand:

- Volume of system log information output since the previous synchronization point
- Amount of time that has elapsed since the previous synchronization point

***system-log-output-volume*: ~ <unsigned integer>((100-100000)) (Number of log blocks)**

Specifies an interval between synchronization point dumps in terms of the number of blocks of log information. A synchronization point dump is collected each time system log information equivalent to the number of log blocks specified here has been output.

***interval*: ~ <unsigned integer>((0 or 10-1440)) (Minutes)**

Specifies a synchronization point dump collection interval in terms of number of minutes between synchronization point dumps.

- When `0` is specified for the interval, HiRDB does not use a time interval for collecting synchronization point dumps.
- If no transactions execute during an interval, no synchronization point dump is collected even though the amount of the time specified here has elapsed.

Specification guidelines

- This operand need not be specified if no specific amount of time is specified for restarting HiRDB.
- The value specified for this operand affects the amount of time required to restart HiRDB.
- Specifying a small value for this operand reduces the amount of time required for database recovery during a HiRDB restart. However, because the frequency of synchronization point dumps increases, online performance may deteriorate in some cases.

  Conversely, specifying a large value for this operand increases the amount of time required for database recovery during a HiRDB restart. However, because the frequency of synchronization point dumps decreases, online performance may improve in some cases.

Tuning the specified value

The synchronization point dump collection interval can be checked with the statistics analysis utility; the relevant information is shown under `SYNC POINT GET INTERVAL` in the statistical information related to system operation. The average `SYNC POINT GET INTERVAL` value should be used. If the synchronization point dump collection interval is determined to be too long, the specification value should be decreased; conversely, if it is determined to be too short, the specification value should be increased.

Note

The synchronization point dump collection interval is determined on the basis of the volume of system log information that is output. Therefore, committing data from memory to a database takes a long time during intervals that have few updating transactions. If an error occurs at such a time, it will take longer to recover the transactions that were generated during that period. If this is a possibility, the synchronization point dump collection interval should be set by also using the `interval` value.

5.3.18 Operands related to server status files

59) pd_sts_file_name_1 = "*logical-file-name*", "*file-a-status-file-name*", "*file-b-status-file-name*"

: . . .

pd_sts_file_name_7 = "*logical-file-name*", "*file-a-status-file-name*", "*file-b-status-file-name*"

Define server status files. Although the `pd_sts_file_name_2` to `7` operands may be omitted, the `pd_sts_file_name_1` operand cannot be omitted.

"*logical-file-name*" ～ <identifier>((1-8 characters))

Specifies the logical file name of a status file for the single server.

When a command that manipulates the status file is to be executed, the logical file name defined here must be specified.

"*file-a-status-file-name*" ～ <pathname>((up to 167 characters))

Specifies the name of the File A status file as an absolute pathname.

"*file-b-status-file-name*" ～ <pathname>((up to 167 characters))

Specifies the name of the File B status file as an absolute pathname.

Specification guidelines

- The files specified as File A and File B should be status files created with the `pdstsinit` command.
- If a file that has not been created with the `pdstsinit` command is specified, a virtual status file is created.
- If an error occurs in a status file, HiRDB swaps the status files. If no spare file is available for swapping, the unit terminates abnormally. For this reason, defining a large number of system files improves system reliability, but at the expense of increasing the amount of disk space that is required.
- The status files specified for File A and File B must have the same record length and capacity.

Operand rules

- Up to seven instances of this operand may be specified.
- A status file has a dual structure consisting of File A and File B; both must be specified.
- Environment variables cannot be used for the absolute pathnames of the File A and File B file names.

- The same names cannot be specified for the logical file name, the File A file name, and the File B file name.

Notes

- When HiRDB is started normally, the primary file (the file that was the primary file when HiRDB was terminated) is inherited. However, if there is no current file that can be inherited, for example because all status files have been initialized, the first status file that was specified among those specified by the `pd_sts_file_name_1` to `7` operands becomes the current file, and the remaining files become spare files if they can be opened. Any files that cannot be opened become reserved files.
- When HiRDB is restarted, the primary file (the file that was the primary file when HiRDB was terminated) is inherited.

Using a virtual status file

Specifying a virtual status file makes it possible to add a new status file while HiRDB is running.

For example, if the number of spare files is reduced because of errors in status files, virtual status files can be converted into spare files. The procedure for converting virtual status files into spare files follows.

Procedure

1. Use the `pdstsinit` command to create a status file in a HiRDB file system area for system files.
2. Use the `pdstsopen` command to open the status file.

These operations can be performed while HiRDB is running; there is no need to stop HiRDB.

- **Advantages and disadvantages**

  Defining a virtual status file reduces the amount of space required in the HiRDB file system area. However, a virtual status file cannot be added as a spare file if the HiRDB file system area for system files does not have sufficient space (sufficient space for adding the file), thus reducing system reliability.

  When virtual status files are not defined, the amount of space required in the HiRDB file system area is increased. However, because a swap destination is guaranteed when a file error occurs, system reliability is increased.

- **Notes**

  When virtual status files are defined, HiRDB determines that a status file error has occurred when the unit is started. For this reason, HiRDB cannot start if `stop` (default value) is specified for the `pd_sts_initial_error` operand. When virtual status files are defined, specify `continue` or `excontinue` for the `pd_sts_initial_error` operand. It is also necessary before starting HiRDB to specify the current file in the `pd_sts_last_active_file` operand.

Relationship to other operands

This operand is related to the `pd_sts_subfile_name_1` to `7` operands.

60) pd_sts_subfile_name_1 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"

:

pd_sts_subfile_name_7 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"

Defines the server status files for log application processing, which are used at the log application site when Real Time SAN Replication based on the log-only synchronous method is used.

"*logical-file-name*": ~ <identifier> ((1-8 characters)

Specifies the logical file name of a server status file for log application processing. This must be a logical file name specified in `pd_sts_file_name_1` through `pd_sts_file_name_7`. A logical file name defined here is used for executing commands that manipulate the status files for log application processing.

"*primary-status-file-name-for-log-application-processing*": ~ <path name> ((up to 167 characters)

Specifies the absolute path name of a primary status file for log application processing.

"*secondary-status-file-name-for-log-application-processing*": ~ <path name> ((up to 167 characters)

Specifies the absolute path name of a secondary status file for log application processing.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync`

must be specified in the `pd_rise_pairvolume_combination` operand.

Specification guidelines

- For the primary and secondary status file names for log application processing, specify the names of status files for log application processing that were created in the preparations for log application. If any other status file for log application processing is specified, the specified file becomes a *virtual status file for log application processing*.
- If an error occurs in a status file for log application processing, HiRDB swaps status files for log application processing. If no spare file is available for swapping, HiRDB (or the unit in the case of a HiRDB/ Parallel Server) terminates abnormally. For this reason, defining a large number of status files improves system reliability, but at the expense of increasing the amount of disk space that is required.
- The specified primary and secondary status files for log application processing must have the same record length and capacity.

Operand rules

- You can specify a maximum of 7 instances of this operand.
- This operand is ignored at the transaction execution site.
- If you performed the preparations for log application during HiRDB startup, create status files for log application processing that correspond to all status files for transaction processing that can be opened and that were specified in `pd_sts_file_name_1` through `pd_sts_file_name_7`.
- The system uses dual status files for log application processing. Make sure that you specify both files.
- Environment variables cannot be used for the absolute path names of the primary and secondary status files for log application processing.
- Each logical file name, primary status file name for log application processing, and secondary status file name for log application processing must be unique.
- The names of the primary and secondary status files for log application processing must be different from the primary and secondary status file names specified in the `pd_sts_file_name_1` through `pd_sts_file_name_7` operands.

Notes

When HiRDB is started normally or is restarted, the current file (the file that was the current file when HiRDB last terminated) is inherited. However, if there is no current file that can be inherited, such as when all status files for log application processing have been initialized, the log application site can no longer be started. If this occurs, perform the preparations for log application.

How to use a virtual status file for log application processing

See the description of the `pd_sts_file_name_1` through `pd_sts_file_name_7` operands.

Relationship to other operands

This operand is related to the `pd_sts_file_name_1` through `pd_sts_file_name_7` operands.

5.3.19 Operands related to server status files (when an error occurs)

For details about the measures to be taken when an error occurs in a status file, see the *HiRDB Version 8 System Operation Guide*.

61) pd_sts_initial_error = stop | continue | excontinue

When a server (single server, front-end server, dictionary server, or back-end server) starts, HiRDB performs a process of identifying the current server status file. This operand specifies the action that HiRDB takes when any of the following errors are detected during this identification process.

- No real server status file is found.
- An error is detected in the server status file.

The current file identification process is applied to the server status files specified by the `pd_sts_file_name_1` to `7` operands.

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing specified in the `pd_sts_subfile_name1` through `pd_sts_subfile_name7` operands are also used at the log application site. For the log application site, replace the `pd_sts_last_active_file` and `pd_sts_last_active_side` operands in the above description with the `pd_sts_last_active_subfile` and `pd_sts_last_active_side_sub` operands, respectively.

`stop`:

When an error is detected in a server status file during the current file identification process, the startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started. In this case, first take a corrective action for the status file in which the error was detected, and then start HiRDB.

continue or excontinue:

Even when an error is detected in the server status file during the current file identification process, the startup of the server is continued if the current file is normal. However, the startup may be stopped depending on the value specified for the pd_sts_singleoperation operand (whether operation should continue with a single status file). The following table shows the relationship to the pd_sts_singleoperation operand.

- **Relationship to the pd_sts_singleoperation operand**

pd_sts_singleoperation operand value	Processing by HiRDB	HiRDB administrator's action
continue	When an error is detected in a server status file, HiRDB cannot identify the current file, and thus startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.	The HiRDB administrator identifies the current file and specifies the pd_sts_last_active_file and pd_sts_last_active_side operands. Afterwards, start HiRDB.
stop (default value)	When an error is detected in a server status file, HiRDB identifies the current file and startup of the server is continued. However, if the primary and secondary files satisfy any of the conditions listed in the table below (cases in which HiRDB cannot identify the current file), startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.	If HiRDB cannot identify the current file, the HiRDB administrator identifies the current file and specifies the pd_sts_last_active_file and pd_sts_last_active_side operands. Afterwards, start HiRDB.

- **When HiRDB cannot identify the current file**

pd_sts_initial_error operand value	File A status	File B status
`continue`	Error shutdown	Error shutdown
	Error shutdown	Open (initial state)
	Error shutdown	No real file
	Open (initial state)	Error shutdown
	Open (initial state)	No real file
	No real file	Open (initial state)
	No real file	Error shutdown
	No real file	No real file
`excontinue`	Error shutdown	Error shutdown
	Error shutdown	No real file
	No real file	Error shutdown
	No real file	No real file

Therefore, to minimize actions by the HiRDB administrator (to reduce the number of cases in which both the `pd_sts_last_active_file` and `pd_sts_last_active_side` operands must be specified), specify the following:

- `pd_sts_initial_error = excontinue`
- `pd_sts_singleoperation = stop`

Specification guidelines

The following table shows the specification guidelines, along with their advantages and disadvantages.

Item	pd_sts_initial_error operand value	
	stop	continue or excontinue
Processing by HiRDB during server startup	When an error is detected in a server status file, startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/ Parallel Server) is not started.	Even when an error is detected in a server status file, the startup of the server is continued if the current file is normal.

Item	pd_sts_initial_error operand value	
	stop	continue or excontinue
Specification guideline	To improve system reliability, specify `stop`.	To simplify the error-handling actions during HiRDB startup, specify `continue` or `excontinue`.
Advantage	Guarantees that all server status files of the server are normal when the server starts. Therefore, if an error occurs in the current file after HiRDB has started, it can be swapped to a spare file.	Even when an error is detected in a server status file during server startup, HiRDB can start with the remaining normal files only. Therefore, HiRDB stop time can be shortened. In this case, because the number of spare files has become small, it is necessary to immediately repair the status files containing errors.
Disadvantage	Possibility increases that an error in a server status file stops the startup of HiRDB.	Because HiRDB may be running with only a small number of spare files, system reliability is low. Depending on the number of spare files available, it may not be possible to swap server status files.

Notes

- If both current files are abnormal, startup of the server is stopped regardless of the value specified for this operand, and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.
- Before starting HiRDB, do not initialize the current file with the `pdstsinit` command. If the current file is initialized, HiRDB cannot be restarted.

Remarks

For details about the operand value, HiRDB processing, and HiRDB administrator's action, see *Remarks* in the section on the `pd_sts_initial_error` operand.

62) pd_sts_singleoperation = stop | continue

Specifies whether or not processing of server status files may continue in the single operation mode. The status file single-operation mode means that processing is to continue using only the normal file (single file) when an error occurs in the status file and a spare file is not available. For details about the status file single-operation mode, see the *HiRDB Version 8 Installation and Design Guide*.

If an error occurs in one of the current files and a spare file is available, the status file is swapped and processing continues regardless of the value specified for this operand (operation in the single-operation mode does not occur).

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing are also used at the log application site.

`stop`:

Do not permit operation in the single-operation mode (the unit is to be terminated abnormally instead). If HiRDB is abnormally terminated, allocate spare files and then restart HiRDB.

`continue`:

Enable operation in the single-operation mode (processing is to continue using the normal side of the current file). When the single-operation mode goes into effect, the message `KFPS01044-I` is output. If an error occurs in the normal file during operation in the single-operation mode or if HiRDB is abnormally terminated while the status file is being updated, HiRDB cannot be restarted. Therefore, when the single-operation mode goes into effect, immediately allocate spare files.

Specification guidelines

- Hitachi recommends that you specify `stop` to increase system reliability. Hitachi also recommends that you increase the number of spare files to guard against errors in the current files.
- The following table shows the specification guidelines, along with their advantages and disadvantages.

Item	pd_sts_singleoperation operand value	
	stop	continue
Specification guideline	To improve system reliability, specify `stop`.	Specify `continue` if it is important not to stop HiRDB.
Advantage	When an error occurs in one of the current files and a spare file is not available, operation in the single-operation mode does not occur and HiRDB is abnormally terminated. Consequently, the possibility of losing the content of the current files is reduced.	Even when an error occurs in one of the current files and a spare file is not available, processing can be continued. Therefore, the possibility that an error in the status file stops HiRDB is reduced.
Disadvantage	Possibility that an error in the status file stops HiRDB increases. However, increasing the number of spare files can reduce this possibility.	If an error occurs in the normal status file during operation in the single-operation mode or if HiRDB is abnormally terminated during updating of the status file, the content of the current file is lost and HiRDB cannot be restarted.

Relationship to other operands

The combination of the values specified for the pd_sts_singleoperation and pd_sts_initial_error operands determines the processing by HiRDB when an error occurs in the status file. Therefore, the values to be specified for these operands should be determined together.

63) pd_sts_last_active_file = "*logical-file-name*"

~ <identifier>((1-8 characters))

Specifies the name of the logical file to be used as the current status file at the time of HiRDB startup. HiRDB compares the file name specified in this operand with the file name selected by HiRDB to be the current file. If the file names match, HiRDB is started; otherwise, HiRDB is not started.

Conditions

The following conditions must be satisfied:

- continue or excontinue is specified in the pd_sts_initial_error operand.
- It cannot be determined if the current file selected by the HiRDB system was the most recent current file during the previous session.

Specification guidelines

1. To start HiRDB immediately after initializing all status files:

   From among the operable logical files specified in the pd_sts_file_name1-7 operands, specify the one that has the smallest number. Forced startup will be used in this case, regardless of the previous termination mode.

2. When both of the current files are normal:

   Specify the name of the current file.* If HiRDB cannot be started even though the name of the current file is specified, the current file may have been initialized. In this case, first initialize all status files, then use the method in 1 above to start HiRDB (forced startup will be used, regardless of the previous termination mode).

3. When one of the current status files has an error:

   Use the method in 2 previously, with the following operands specified:

   - pd_syssts_singleoperation = continue
   - pd_sts_last_active_side

4. When both of the current status files have errors:

Initialize all status files, then execute the method in 1 above (forced startup will be used, regardless of the previous termination mode).

5. When a virtual status file is specified

   Specify the name of the current file.*

* The names of the current files (that were active at the end of the previous operation) can be determined from the following messages:

- `KFPS01001-I`
- `KFPS01010-E`
- `KFPS01011-I`
- `KFPS01063-I`

Of the status files displayed by these messages, the one that is reported in the message that was output most recently is the current file.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for transaction processing are used at the transaction execution site. At the log application site, this operand is ignored, if specified.

64) pd_sts_last_active_side = A | B

Specify this operand if you want to start HiRDB when one of the current files is in an error state. Specify the normal status file for this operand. HiRDB compares the file specified in this operand with the file selected by HiRDB. If they match, HiRDB copies the contents of the normal status file to secondary File A and File B, then HiRDB switches the spare file in as the current file and starts the unit. If the files do not match, HiRDB is not started.

Conditions

The following operands must be specified:

- `pd_sts_initial_error = continue` or `excontinue`
- `pd_sts_last_active_file`

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for transaction processing are used at the transaction execution site. At the log application site, this operand is ignored, if specified.

65) pd_sts_last_active_subfile = **"*logical-file-name*"**

~ <identifier> ((1-8 characters))

When Real Time SAN Replication based on the log-only synchronous method is used, this operand specifies the logical file name of the status file for log application processing that is to be used as the current file when HiRDB (or the unit in the case of a HiRDB/Parallel Server) starts. HiRDB compares the file specified in this operand with the current file selected by HiRDB. If the file names match, HiRDB starts; if not, HiRDB does not start. Note that this operand is ignored if HiRDB is started immediately after the database is inherited pursuant to execution of the `pdrisedbto` command.

Conditions

For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*

The following conditions must be satisfied:

- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.
- `continue` or `excontinue` is specified in the `pd_sts_initial_error` operand.
- It cannot be determined if the current file selected by the HiRDB system was the most recent current file during the previous session.

Specification guidelines

1. When starting HiRDB immediately after initializing all status files for log application processing

   Perform the preparations for log application. Once the preparations for log application have been completed successfully, start HiRDB without specifying this operand.

2. When both of the current files are normal

   Specify the name of the current file.[#] If HiRDB cannot be started even though the name of the current file is specified, the current file may have been initialized. In such a case, first initialize all status files for log application processing, then use the method in 1 above to start HiRDB. Note that in this case, forced startup will be used, regardless of the previous termination mode.

3. When one of the current files has an error

   Use the method in 2 above and specify the following operands:

   - Specify `continue` in the `pd_syssts_singleoperation` operand

- Specify the `pd_sts_last_active_side_sub` operand

4. When both of the current files have errors

   Perform the preparations for log application. Once the preparations for log application have been completed successfully, start HiRDB without specifying this operand.

5. When a virtual status file for log application processing is specified

   Specify the name of the current file.#

#

The names of the current files (that were active at the end of the previous session) can be determined from the following messages:

- `KFPS01001-I`
- `KFPS01010-E`
- `KFPS01011-I`
- `KFPS01063-I`

Of the status files for log application processing that are displayed in these messages, the one that is reported in the message that was output most recently is the current file.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing are used at the log application site. At the transaction execution site, this operand is ignored, if specified.

66) pd_sts_last_active_side_sub = A | B

When Real Time SAN Replication based on the log-only synchronous method is used, specify this operand if you wish to start HiRDB (or the unit in the case of a HiRDB/Parallel Server) while one of the current files is in error status. In such a case, you specify the normal file in this operand. HiRDB compares the file specified in this operand with the file selected by HiRDB. If their names match, HiRDB copies the contents of the normal file into the secondary file. HiRDB then designates the secondary file as the current file and starts itself. If the file names do not match, HiRDB does not start. This operand is ignored if HiRDB is started immediately after the database was inherited pursuant to execution of the `pdrisedbto` command.

Conditions

For details about the required platform, see the *HiRDB Version 8 Disaster*

Recovery System Configuration and Operation Guide.

The following conditions must be satisfied:

- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.
- `continue` or `excontinue` is specified in the `pd_sts_initial_error` operand.
- The `pd_sts_last_active_subfile` operand is specified.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing are used at the log application site. At the transaction execution site, this operand is ignored, if specified.

5.3.20 Operands related to work table files

67) pdwork -v "*HiRDB-file-system-area-name*"[,"*HiRDB-file-system-area-name*"]...

～ <pathname of up to 141 characters>

This operand must be specified in order to execute SQL statements.

Specifies the names of HiRDB file system areas for work table files. Work table files are used for temporary storage of information during execution of SQL statements; they are created automatically by HiRDB. For details about the SQL statements that require a work table file, see the *HiRDB Version 8 Description*.

Notes

- Specify in this operand the HiRDB file system area that was initialized using the `pdfmkfs` command.
- If the size of the work table file is large, specify a large HiRDB file system area. For the method of estimating the work table file size, see the *HiRDB Version 8 Installation and Design Guide*.
- The HiRDB file system areas for work table files must be different from the HiRDB file system areas for system files and RDAREAs.
- In the Linux version, you cannot specify a HiRDB file system area created in a character special file.

Operand rules

- At least one HiRDB file system area must be specified.

- A maximum of 16 HiRDB file system areas may be specified.
- This operand may be specified only once in the single server definition. If it is specified more than once, the first specification is effective.
- When you are specifying multiple HiRDB file system areas, those created in character special files can coexist with those created in regular files.

5.3.21 Operands related to system log file configuration

68) pdlogadfg -d sys -g *file-group-name* [ONL]

Specifies a file group for a system log file. This operand cannot be omitted and must be specified. Use the `pdlogadpf` operand to assign system log files to the file group specified here.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name* ~ <identifier>((1-8 characters))

Specifies a name for the file group. All file group names must be unique within a server.

`ONL`:

Specify this option to enable (open) this file group when HiRDB is running. You can specify 2 to 200 file groups with `ONL` specified.

Operand rule

This operand must be specified at least twice but no more than 200 times.

Notes

Before adding, modifying, or deleting this operand when HiRDB Datareplicator (extracted side) is linked, terminate the corresponding HiRDB Datareplicator. If this operand is modified while HiRDB Datareplicator is running, its extraction process may fail.

69) pdlogadpf -d sys -g *file-group-name* -a "*system-log-file-name*" [-b "*system-log-file-name*"]

Specifies the system log files that comprise the file group. This operand cannot be omitted; it must be specified once for each file group.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name* ~ <identifier>((1-8 characters))

Specifies the file group name specified by the `pdlogadfg` operand. The file group name must be unique within the unit.

-a "*system-log-file-name*" ~ <pathname>((up to 167 characters))

Specifies an absolute path name as the name of the system log file that comprises the file group. Specify the name of the system log file that was initialized using the `pdloginit` command. Note that the system log file name must be unique within the unit.

-b "*system-log-file-name*" ~ <pathname>((up to 167 characters))

Specifies an absolute path name as the name of the system log file B when dual system log files are to be used (`pd_log_dual = Y` specified). If `pd_log_dual = Y` is not specified, the system log file name is invalid, even if it is specified.

Specify the name of the system log file that was initialized using the `pdloginit` command. Note that the system log file name must be unique within the unit.

Note

Before adding, modifying, or deleting this operand when HiRDB Datareplicator (extracted side) is linked, terminate the corresponding HiRDB Datareplicator. If this operand is modified while HiRDB Datareplicator is running, its extraction process may fail.

5.3.22 Operands related to synchronization point dump file configuration

70) pdlogadfg -d spd -g *file-group-name* [ONL]

Specifies a file group for synchronization point dump files. This operand cannot be omitted and must be specified. Use the `pdlogadpf` operand to assign synchronization point dump files to the file group specified here.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name* ~ <<identifier>((1-8 characters))

Specifies a name for the file group. All file group names must be unique within a server.

`ONL`:

Specify this option to enable (open) this file group when HiRDB is running. You can specify 2 to 30 file groups with `ONL` specified.

Operand rule

This operand must be specified at least twice but no more than 60 times.

71) pdlogadpf -d spd -g *file-group-name* -a "*synchronization-point-dump-file-name*" [-b "*synchronization-point-dump-file-name*"]

Specifies the synchronization point dump files that comprise the file group. This operand cannot be omitted; it must be specified once for each file group.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name*: ~ <identifier>((1-8 characters))

Specifies the file group name specified by the `pdlogadfg` operand. The file group name must be unique within the unit.

-a "*synchronization-point-dump-file-name*": ~ <pathname> ((up to 167 characters))

Specifies an absolute path name as the name of the synchronization point dump file that comprises the file group. Specify the name of the synchronization point dump file that was initialized using the `pdloginit` command. Note that the synchronization point dump file name must be unique within the unit.

-b "*synchronization-point-dump-file-name*": ~ <pathname> ((up to 167 characters))

Specifies an absolute path name as the name of the synchronization point dump file B when dual synchronization point dump files are to be used (`pd_spd_dual = Y` specified). If `pd_spd_dual = Y` is not specified, the synchronization point dump file name is invalid, even if it is specified.

Specify the name of the synchronization point dump file that was initialized using the `pdloginit` command. Note that the synchronization point dump file name must be unique within the unit.

72) pdlogadfg -d ssp -g *file-group-name*[ONL]

When Real Time SAN Replication based on the log-only synchronous method is used, specifies a group of synchronization point dump files for log application processing that is to be used at the log application site. The `pdlogadpf` operand is used to allocate the synchronization point dump files for log application processing.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name*: ~ <identifier> ((1-8 characters))

Specifies the name of a file group. This file group name must correspond to a set of synchronization point dump files for transaction processing.

`ONL`:

Specifies that this file group is to be made available (opened) when HiRDB is running. You can specify 2 to 30 file groups with `ONL` specified.

If Real Time SAN Replication based on the log-only synchronous method is used, when HiRDB is started for the first time at the log application site after the preparations for log application have been performed, the file groups that need to be inherited from the transaction execution site become available when the single server starts, regardless of the specification of `ONL`.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.

Operand rules

You can specify 2 to 60 instances of this operand.

Notes

Specify a `pdlogadfg -d ssp` operand for each file group specified in a `pdlogadfg -d spd` operand.

73) pdlogadpf -d ssp -g *file-group-name* -a "*synchronization-point-dump-file-name-for-log-application-processing*" [-b "*synchronization-point-dump-file-name-for-log-application-processing*"]

Specifies synchronization point dump files for log application processing that constitute a file group. You must specify one `pdlogadpf -d ssp` operand for each file group.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name*: ~ <identifier> ((1-8 characters))

Specifies the name of a file group specified in a `pdlogadfg` operand. This file group name must correspond to a set of primary synchronization point dump files for transaction processing.

-a "*synchronization-point-dump-file-name-for-log-application-processing*": ~ <path name> ((up to 167 characters))

Specifies the absolute path name of a synchronization point dump file for log application processing that is to be included in the file group. This must be the name of a synchronization point dump file for log application processing that was created during the preparations for log application. Note that each synchronization point dump file name for log application processing must be unique within the same unit.

-b "*synchronization-point-dump-file-name-for-log-application-processing*": ~ <path name> ((up to 167 characters))

When dual synchronization point dump files for log application processing are used (`Y` is specified in the `pd_spd_dual` operand), specifies the absolute path name of a secondary synchronization point dump file for log application processing. If `Y` is not specified in the `pd_spd_dual` operand, the `-b` option is ignored, if specified. You must specify in this option the name of a synchronization point dump file for log application processing created during the preparations for log application. Note that each synchronization point dump file for log application processing must be unique within the same unit.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.

Notes

Specify a `pdlogadpf -d ssp` operand for each file group specified in a `pdlogadfg -d ssp` operand.

5.3.23 Operands related to Plug-ins

74) pdplgprm -n *plug-in-name* [-s *shared-memory-size*]

Specifies the name of a plug-in and the size of the memory to be shared by the plug-in. This operand should be omitted if no plug-ins are to be used.

Conditions

The plug-in specified here must have been registered in HiRDB with the `pdplgrgst` command.

`-n` *plug-in-name*: ~ <identifier>((1-30 characters))

For details about the names of plug-ins that can be specified here, see the manuals for the plug-ins.

-s *shared-memory-size*: ~ <unsigned integer>((1-2000000))<<0>> (KB)

Specifies in kilobytes the size of the shared memory to be used by the plug-in. For details about the size of the shared memory to be used by the plug-in, see the manual for the applicable plug-in.

Chapter

6. Front-End Server Definition

This chapter explains the operands of the front-end server definition.

This chapter contains the following sections:

6.1 Operand formats

A front-end server definition defines information for a front-end server. This section explains the formats used to specify the operands of a front-end server definition. Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *6.2 Operands whose specification values can be changed* and *6.3 Operand explanations*.

For users who are creating HiRDB system definitions for the first time

The first step is to determine the values to be specified for the operands shown in boldface type. In principle, HiRDB can be started once the boldface operands have been specified.

No.	Format	Operand category
1	[set pd_process_count = *resident-processes-count*[,*resident-processes-count-at-server-startup*]]*	Processes
2	[set pd_server_cleanup_interval = *interval-for-stopping-nonresident-server-processes*]*	
3	[set pd_svr_castoff_size = *maximum-memory-size-used-by-server-process*]*	
4	[set pd_optimize_level = *SQL-optimization-option* [,*SQL-optimization-option*]...]*	SQL optimization
5	[set pd_additional_optimize_level = *SQL-extension-optimizing-option* [,*SQL-extension-optimizing-option*]...]*	
6	[set pd_floatable_bes = "*back-end-server-name*"[,"*back-end-server-name*"]...]	
7	[set pd_non_floatable_bes = "*back-end-server-name*"[,"*back-end-server-name*"]...]	
8	[set pd_watch_pc_client_time = *maximum-client-request-wait-time*]*	System monitoring
9	[set pd_spd_syncpoint_skip_limit = *maximum-number-of-skipped-synchronization-point-dumps*]*	
10	[set pd_cwaittime_wrn_pnt = *output-condition-for-SQL-runtime-warning-information (% specification)* | *output-condition-for-SQL-runtime-warning-information (time specification)*]*	SQL runtime warning output facility

No.	Format	Operand category
11	`[set pd_uap_exerror_log_use = YES	NO]`	Facility for output of extended SQL error information
12	`[set pd_fes_lck_pool_size =` *front-end-server-lock-pool-size*`]`*	Lock
13	`[set pd_lck_hash_entry =` *lock-pool-hash-entry-count*`]`*	
14	`[set pd_sql_object_cache_size =` *SQL-object-buffer-size*`]`*	Buffers
15	`[set pd_table_def_cache_size =` *table-definition-information-buffer-size*`]`*	
16	`[set pd_auth_cache_size =` *user-privilege-information-buffer-size*`]`*	
17	`[set pd_view_def_cache_size =` *view-analysis-information-buffer-size*`]`*	
18	`[set pd_alias_cache_size =` *table-alias-definition-information-buffer-size*`]`*	
19	`[set pd_type_def_cache_size =` *user-defined-type-information-buffer-size*`]`*	
20	`[set pd_routine_def_cache_size =` *routine-definition-information-buffer-size*`]`*	
21	`[set pd_registry_cache_size =` *registry-information-buffer-size*`]`*	
22	`[set pd_rpc_trace = Y	N]`*	RPC trace information
23	`[set pd_rpc_trace_name = "`*name-for-RPC-trace-collection-files*`"]`*	
24	`[set pd_rpc_trace_size =` *RPC-trace-collection-file-size*`]`*	
25	`[set pd_module_trace_max =` *maximum-number-of-module-traces-that-can-be-stored*`]`*	Troubleshooting information
26	`[set pd_module_trace_timer_level =` *module-trace-output-time-acquisition-method*`]`	

No.	Format	Operand category
27	`[set pd_audit_def_buffer_size =` *security-audit-information-buffer-length*`]`	Security audit facility
28	`[set pd_node_name =` *local-node-RD-node-name*`]`	Distributed database
29	`[set pd_java_stdout_file =` "*Java-virtual-machine-standard-output-and-standard-error-output-destination-file*"`]`*	Java
30	`[set pd_log_dual = Y | N]`*	System log files
31	`[set pd_log_dual_write_method = serial | parallel]`	
32	`[set pd_log_remain_space_check = warn | safe]`*	
33	`[set pd_log_auto_unload_path =` "*unload-log-file-output-directory*" *[,*"*unload-log-file-output-directory*"`]`...`]`	
34	`[set pd_log_singleoperation = Y | N]`*	
35	`[set pd_log_rerun_reserved_file_open = Y | N]`*	
36	`[set pd_log_rerun_swap = Y | N]`*	
37	`[set pd_log_swap_timeout =` *wait-time-for-completion-of-system-log-file-swapping*`]`*	
38	`[set pd_log_unload_check = Y | N]`*	
39	`[set pd_log_max_data_size =` *log-input/output-buffer-size*`]`*	
40	`[set pd_log_write_buff_count =` *log-output-buffer-sectors-count*`]`*	
41	`[set pd_log_rec_leng =` *system-log-file-record-length*`]`*	

No.	Format	Operand category
42	[set pd_spd_dual = Y | N]*	Synchronization point dump files
43	[set pd_spd_assurance_msg = Y | N]*	
44	[set pd_spd_assurance_count = *number-of-guaranteed-valid-generations*]*	
45	[set pd_spd_reduced_mode = *reduced-mode-operation-option*]*	
46	[set pd_spd_reserved_file_auto_open = Y | N]*	
47	[set pd_spd_max_data_size = *synchronization-point-dump-file-buffer-size*]*	
48	[set pd_log_sdinterval = *system-log-output-volume*[,*interval*]]*	
49	**set pd_sts_file_name_1 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"**	Server status files
	[set pd_sts_file_name_2 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_3 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_4 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_5 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_6 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_7 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
50	[set pd_sts_subfile_name_1 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_2 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	

No.	Format	Operand category
	[set pd_sts_subfile_name_3 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_4 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_5 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_6 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_7 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
51	[set pd_sts_initial_error = stop | continue | excontinue]*	Server status files (when an error occurs)
52	[set pd_sts_singleoperation = stop | continue]*	
53	[set pd_sts_last_active_file = "*logical-file-name*"]	
54	[set pd_sts_last_active_side = A | B]	
55	[set pd_sts_last_active_subfile = "*logical-file-name*"]	
56	[set pd_sts_last_active_side_sub = A | B]	
57	**{{pdlogadfg -d sys -g *file-group-name* [ONL]}}**	System log file configuration
58	**{{pdlogadpf -d sys -g *file-group-name* -a "*system-log-file-name*" [-b "*system-log-file-name*"]}}**	
59	**{{pdlogadfg -d spd -g *file-group-name* [ONL]}}**	Synchronization point dump file configuration

No.	Format	Operand category
60	**{{pdlogadpf -d spd -g *file-group-name* -a "*synchronization-point-dump-file-name*" [-b "*synchronization-point-dump-file-name*"]}}**	
61	{{pdlogadfg -d ssp -g file-group-name [ONL]}}	
62	{{pdlogadpf -d ssp -g *file-group-name* -a "*synchronization-point-dump-file-name-for-log-application-processing*" [-b "*synchronization-point-dump-file-name-for-log-application-processing*"]}}	
63	{{[pdplgprm -n *plug-in-name* [-s *shared-memory-size*]]}}	Plug-ins

64	{
	[pdhubopt -s *foreign-server-name* -f *Hub-optimization-information-definition-file-name*]
	|
	[pdhubopt -d *foreign-server-type* [-v *foreign -server-version*] -f *Hub-optimization-information-definition-file-name*]
	}
		HiRDB External Data Access facility

* If this operand is omitted, the value specified in the same operand in the server common definition is used. However, for the following operands, the value specified for the same operand in the system common definition, rather than the server common definition, is used:

- pd_optimize_level
- pd_additional_optimize_level
- pd_cwaittime_wrn_pnt

6.2 Operands whose specification values can be changed

The values of some of the front-end server definition operands can be changed in the system common definition, unit control information definition, and server common definition. These operands are indicated below. After a planned termination, forced termination, or abnormal termination of HiRDB, some HiRDB system definition operands can be modified while others cannot be modified. The operands that can be modified are indicated below.

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
1	pd_process_count	—	—	Y	Y	Y	Y	Y[2]	Y[2]
2	pd_server_cleanup_interval	—	—	Y	Y	Y	Y	Y	Y
3	pd_svr_castoff_size	—	—	Y	Y	Y	Y	Y	Y
4	pd_optimize_level[4]	Y	—	—	Y	—	—	Y	Y
5	pd_additional_optimize_level[4]	Y	—	—	Y	—	—	Y	Y
6	pd_floatable_bes	—	—	—	Y	—	—	Y	Y
7	pd_non_floatable_bes	—	—	—	Y	—	—	Y	Y
8	pd_watch_pc_client_time[4]	—	—	Y	Y	—	—	Y	Y
9	pd_spd_syncpoint_skip_limit	—	—	Y	Y	Y	Y	Y	Y
10	pd_cwaittime_wrn_pnt[4]	Y	—	—	Y	—	—	Y	Y

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
11	pd_uap_exerror_log_use[4]	Y	—	—	Y	—	—	Y	Y
12	pd_fes_lck_pool_size	—	—	Y	Y	—	—	Y	Y
13	pd_lck_hash_entry	—	—	Y	Y	Y	Y	Y	Y
14	pd_sql_object_cache_size	Y	—	Y	Y	Y	Y	N	Y
15	pd_table_def_cache_size	—	—	Y	Y	—	—	Y[2,3]	Y[2]
16	pd_auth_cache_size	—	—	Y	Y	—	—	Y[2,3]	Y[2]
17	pd_view_def_cache_size	—	—	Y	Y	—	—	Y[2,3]	Y[2]
18	pd_alias_cache_size	—	—	Y	Y	—	—	Y[2,3]	Y[2]
19	pd_type_def_cache_size	—	—	Y	Y	—	—	Y	Y
20	pd_routine_def_cache_size	—	—	Y	Y	—	—	Y	Y
21	pd_registry_cache_size	—	—	Y	Y	—	—	Y	Y
22	pd_rpc_trace	Y	Y	Y	Y	Y	Y	Y	Y
23	pd_rpc_trace_name	Y	Y	Y	Y	Y	Y	Y	Y
24	pd_rpc_trace_size	Y	Y	Y	Y	Y	Y	Y	Y
25	pd_module_trace_max	Y	Y	Y	Y	Y	Y	Y	Y

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
26	`pd_module_trace_timer_level`	Y	Y	Y	Y	Y	Y	Y	Y
27	`pd_audit_def_buffer_size`	—	—	Y	Y	—	Y	Y	Y
28	`pd_node_name`	—	—	—	Y	—	—	Y	Y
29	`pd_java_stdout_file`	Y	Y	Y	Y	Y	Y	Y	Y
30	`pd_log_dual`	—	—	Y	Y	Y	Y	N	N
31	`pd_log_dual_write_method`	—	—	Y	Y	Y	Y	Y	Y
32	`pd_log_remain_space_check`	—	—	Y	Y	Y	Y	Y	Y
33	`pd_log_auto_unload_path`	—	—	—	Y	Y	Y	N	N
34	`pd_log_singleoperation`	—	—	Y	Y	Y	Y	N	N
35	`pd_log_rerun_reserved_file_open`	—	—	Y	Y	Y	Y	Y	Y
36	`pd_log_rerun_swap`	—	—	Y	Y	Y	Y	Y	Y
37	`pd_log_swap_timeout`	—	—	Y	Y	Y	Y	Y	Y
38	`pd_log_unload_check`	—	—	Y	Y	Y	Y	Y	Y
39	`pd_log_max_data_size`	—	—	Y	Y	Y	Y	Y[2,3]	Y[2]

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
40	pd_log_write_buff_count	—	—	Y	Y	Y	Y	Y[3]	Y
41	pd_log_rec_leng	—	—	Y	Y	Y	Y	Y	Y
42	pd_spd_dual	—	—	Y	Y	Y	Y	N	N
43	pd_spd_assurance_msg	—	—	Y	Y	Y	Y	Y	Y
44	pd_spd_assurance_count	—	—	Y	Y	Y	Y	N	N
45	pd_spd_reduced_mode	—	—	Y	Y	Y	Y	N	Y
46	pd_spd_reserved_file_auto_open	—	—	Y	Y	Y	Y	N	Y
47	pd_spd_max_data_size	—	—	Y	Y	Y	Y	N	N
48	pd_log_sdinterval	—	—	Y	Y	Y	Y	Y	Y
49	pd_sts_file_name_1-7	—	—	—	Y	Y	Y	Y[1]	Y[1]
50	pd_sts_subfile_name_1 to 7	—	—	—	Y	Y	Y	N	N
51	pd_sts_initial_error	—	—	Y	Y	Y	Y	Y	Y
52	pd_sts_singleoperation	—	—	Y	Y	Y	Y	Y	Y
53	pd_sts_last_active_file	—	—	—	Y	Y	Y	Y	Y

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
54	`pd_sts_last_active_side`	—	—	—	Y	Y	Y	Y	Y
55	`pd_sts_last_active_subfile`	—	—	—	Y	Y	Y	Y	Y
56	`pd_sts_last_active_side_sub`	—	—	—	Y	Y	Y	Y	Y
57	`pdlogadfg -d sys`	—	—	—	Y	Y	Y	Y[1]	Y[1]
58	`pdlogadpf -d sys`	—	—	—	Y	Y	Y	Y[1]	Y[1]
59	`pdlogadfg -d spd`	—	—	—	Y	Y	Y	Y[1]	Y[1]
60	`pdlogadpf -d spd`	—	—	—	Y	Y	Y	Y[1]	Y[1]
61	`pdlogadfg -d ssp`	—	—	—	Y	Y	Y	Y[1]	Y[1]
62	`pdlogadpf -d ssp`	—	—	—	Y	Y	Y	Y[1]	Y[1]
63	`pdplgprm`	—	—	—	Y	Y	Y	Y	Y
64	`pdhubopt`	—	—	—	Y	—	—	N	Y

Y: Yes, specification value can be modified.

N: No, specification value cannot be modified.

—: Specification value cannot be modified because operand is not applicable.

SYS: System common definition

UNT: Unit control information definition

SVR: Server common definition

FES: Front-end server definition

DS: Dictionary server definition

BES: Back-end server definition

[1] Operand that can be added only; it cannot be deleted or modified.

[2] If the specified value is too small, it may not be possible to restart HiRDB. When this is the case, restore the value to what it was before the change and then restart HiRDB.

[3] If the specified value is too large, it may not be possible to restart HiRDB. When this is the case, restore the value to what it was before the change and then restart HiRDB.

[4] The following values for the HiRDB system definition operands can be different for each client. To change a value for a client, specify the operand in the client environment definition. For details about the client environment definition, see the *HiRDB Version 8 UAP Development Guide*.

HiRDB system definition operand	Client environment definition operand
pd_optimize_level	PDSQLOPTLVL
pd_additional_optimize_level	PDADDITIONALOPTLVL
pd_watch_pc_client_time	PDSWATCHTIME
pd_cwaittime_wrn_pnt	PDCWAITTIMEWRNPNT
pd_uap_exerror_log_use	PDUAPEXERLOGUSE

6.3 Operand explanations

6.3.1 Operands related to processes

1) pd_process_count =
resident-processes-count**[,*****resident-processes-count-at-server-startup*****]**

～ <unsigned integer>((0-2000))

resident-processes-count

Specifies the number of processes that can be made resident in the front-end server. A resident process is a process that is activated at the time the server is started.

Advantage

By activating the processes used by transactions that can be processed concurrently by the front-end server at the time of system startup and keeping them resident, the process startup time can be reduced even when new transactions are entered. However, HiRDB startup will take longer.

Specification guidelines

See *Specification guidelines* for the `pd_process_count` operand of the server common definition.

Tuning the specified value

For details about how to tune the resident processes count, see the *HiRDB Version 8 System Operation Guide*.

Notes

- Because the resident processes count has an effect on memory space availability, specifying an unnecessarily large number may prevent HiRDB from starting.
- If more processes than the resident process count are needed, additional processes are dynamically started, up to the maximum processes count allowed. However, depending on the value specified for the `pd_max_server_process` operand, it may not be possible to start all of the processes indicated by the maximum processes count.

Operand rule

See *Operand rules* for the `pd_process_count` operand of the server common definition.

resident-processes-count-at-server-startup

Specifies the number of processes that can be made resident during HiRDB startup.

It is common for resident processes to be activated during HiRDB startup. If there are many resident processes, the amount of time required for HiRDB startup increases proportionately. For example, it takes approximately 1 second to activate a process on a server machine with a 100-MIPS performance rating.

Following are the differences in processing that result depending on whether or not a resident processes count at server startup is specified:

- When there is no specification of a resident processes count at server startup (when, for example, `pd_process_count = 500` is specified)

  All 500 resident processes are activated during HiRDB startup, and HiRDB will not start until they are all activated. In this example, it would take a 100-MIPS server machine about 500 seconds to activate all the resident processes during HiRDB startup.

- When a resident processes count at server startup is specified (when, for example, `pd_process_count = 500, 50` is specified)

  Some of the resident processes (50 in this case) are activated during HiRDB startup, and the remaining resident processes are activated after HiRDB startup. HiRDB can be started as long as the specified number of resident processes are activated. In this example, it would take a 100-MIPS server machine about 50 seconds to activate 50 resident processes during HiRDB startup. The remaining 450 resident processes (in this example) would be activated after HiRDB startup.

Advantage

The amount of time required for HiRDB startup is reduced. You should use this option when you want to reduce the HiRDB startup time as much as possible, such as when you are using the system switchover facility.

Specification guideline

The specified value should be equal to the number of processes that will be required immediately after HiRDB startup is completed.

Notes

When you specify a resident processes count at server startup, you should recheck the value in the `PDCWAITTIME` operand of the client environment definition.

If more UAPs than the resident processes count at server startup are to be executed immediately following HiRDB startup, transaction processing may not be performed until after the remaining resident processes have been

activated. Therefore, if the value specified in the PDCWAITTIME operand of the client environment definition is small, it may not be possible to process some UAPs due to timeouts. For details about the PDCWAITTIME operand, see the *HiRDB Version 8 UAP Development Guide*.

2) pd_server_cleanup_interval = ***interval-for-stopping-nonresident-server-processes***

~ <unsigned integer>((0-1440)) (minutes)

Specifies in minutes the interval for checking for nonresident server processes in HiRDB that are to be stopped. The facility for stopping nonresident server processes is applied when the number of executing server processes exceeds the number of processes that can be made resident (value specified by the pd_process_count operand). The number of server processes that the facility stops is computed automatically by HiRDB.

Advantages

This facility improves the utilization rates of the memory and of process resources because it increases the number of nonresident processes that can be reused when the workload (number of server processes that are executing) peaks.

Specification guidelines

- For example, if there is only one hour a day during which peak workload occurs and the intervals between peaks within that hour are approximately two minutes apart, 2 should be specified for this operand.
- This facility does not have any effect if the number of server processes that execute concurrently during peak periods is always fewer than the number of resident processes. In this case, this operand should be omitted.

Tuning the specified value

Collect statistical information on system operations for each server for one week. Determine the workload peaks from the number of server processes being serviced (# OF PROCESSES ON SERVICE). If that peak value exceeds the currently set number of processes that can be made resident (value specified by the pd_process_count operand), determine the interval between individual peaks and set that number of minutes.

However, if there are ample resources, such as memory and CPU, in the server machine, adding the shortfall in the number of processes to the number of resident processes (i.e., increasing the value of the pd_process_count operand) would be more effective in improving performance than specifying the pd_server_cleanup_interval

operand.

Note

When this operand is omitted or `0` is specified, the system checks every 10 seconds for nonresident server processes that are waiting to be serviced and stops any such processes that are found.

3) pd_svr_castoff_size = *maximum-memory-size-used-by-server-process*

~ <unsigned integer>((0-2048)) (MB)

This operand need not be specified for the Linux version.

Specifies the maximum size of memory used by each server process processed by the front-end server. If at the applicable trigger point shown below the amount of memory being used by a server process exceeds the value specified here, that server process is terminated. When a server process is so terminated, the `KFPS00350-W` message is output. This is called the *facility for monitoring the memory size of server processes*; for details, see the *HiRDB Version 8 System Operation Guide*.

Server type	Process name	Process termination trigger
Front-end server	pdfes	At UAP disconnection

Advantages

The facility for monitoring the memory size of server processes resolves the following problem:

- *The memory size of a server-resident process becomes too large during a particular SQL processing, significantly reducing the amount of system memory that is available.*

The HiRDB server releases memory space that is no longer needed. However, even when a program releases memory, the OS holds the memory area itself in the memory management facility inside the applicable process. Consequently, a process that becomes large in terms of using a large memory area even once never shrinks and continues to have an adverse impact on the system, especially in the case of resident processes. The facility for monitoring the memory size of server processes can prevent memory shortages because it terminates even resident processes.

Application criterion

Apply this facility when the amount of memory space used by a HiRDB server process becomes large, resulting in memory shortages.

Specification guidelines

- Maximum value for this operand

  Normally specify a value by considering the maximum processing capability of HiRDB. Assuming that the maximum number of SQL statements may be concurrently connected, determine for this operand a value that satisfies the following condition in each unit:

  $a \times (b + c) < d$

  a: Number of server processes within the unit (*maximum-number-of-concurrent-connections* × *server-count-inside-unit*)

  b: Virtual memory size for one server process immediately following HiRDB startup

  c: `pd_svr_castoff_size` operand value

  d: Memory size that can be allocated to a unit (memory size excluding the area used by other programs)

- Minimum value for this operand

  If the value specified for this operand is smaller than the memory size needed for normal SQL processing, the efficiency of making processes resident will deteriorate, resulting in frequent process terminations and restarts. Each time this occurs, a message is output to the syslogfile or the message log file, resulting in further performance degradation. To prevent this, select for this operand a value that satisfies the following condition for each server:

  $a - b < c$

  a: Server process virtual memory size following SQL process termination

  b: Virtual memory size for server process immediately following HiRDB startup

  c: `pd_svr_castoff_size` operand value

The virtual memory size can be determined by an OS command (e.g., the `top` command in HP-UX).

Operand rule

When `0` is specified for this operand, the facility for monitoring the memory size of server processes is not applied.

6.3.2 Operands related to SQL optimization

4) pd_optimize_level = *SQL-optimization-option* [*,SQL-optimization-option*]...

~ <identifier or unsigned integer>

Specifies SQL optimization options. For details about the SQL optimization options, see the *HiRDB Version 8 UAP Development Guide.* The SQL optimization option facilities are explained as follows.

SQL optimization option facility	Identifier	Unsigned integer	S	P
Forced nest-loop-join	`"FORCE_NEST_JOIN"`	4	Y	Y
Making multiple SQL objects	`"SELECT_APSL"`	10	Y	Y
Increasing the target floatable servers (back-end servers for fetching data)	`"FLTS_INC_DATA_BES"`	16	N	Y
Prioritized nest-loop-join	`"PRIOR_NEST_JOIN"`	32	Y	Y
Increasing the number of floatable server candidates	`"FLTS_MAX_NUMBER"`	64	N	Y
Priority of OR multiple index use	`"PRIOR_OR_INDEXES"`	128	Y	Y
Group processing, ORDER BY processing, and DISTINCT set function processing at the local back-end server	`"SORT_DATA_BES"`	256	N	Y
Suppressing use of AND multiple indexes	`"DETER_AND_INDEXES"`	512	Y	Y
Rapid grouping facility	`"RAPID_GROUPING"`	1024	Y	Y
Limiting the target floatable servers (back-end servers for fetching data)	`"FLTS_ONLY_DATA_BES"`	2048	N	Y
Separating data collecting servers	`"FLTS_SEPARATE_COLLECT_SVR"`	2064	N	Y
Suppressing index use (forced table scan)	`"FORCE_TABLE_SCAN"`	4096	Y	Y
Forcing use of multiple indexes	`"FORCE_PLURAL_INDEXES"`	32768	Y	Y
Suppressing creation of update-SQL work tables	`"DETER_WORK_TABLE_FOR_UPDATE"`	131072	Y	Y
Deriving search acceleration condition	`"DERIVATIVE_COND"`	262144	Y	Y
Applying key condition that includes scalar operation	`"APPLY_ENHANCED_KEY_COND"`	524288	Y	Y
Facility for batch acquisition from functions provided by plug-ins	`"PICKUP_MULTIPLE_ROWS_PLUGIN"`	1048576	Y	Y

S: HiRDB/Single Server

P: HiRDB/Parallel Server

Y: Specification is valid.

N: Specification is not applicable.

Operand specification methods

Select the SQL optimization options to be applied and specify their identifiers or unsigned integers. Although either identifiers or unsigned integers (computed values) can be used to specify options, the use of identifiers is usually recommended.

- Using identifiers

  To apply *forced nest-loop-join* and *making multiple SQL objects*, specify the following:

  ```
  pd_optimize_level="FORCE_NEST_JOIN","SELECT_APSL"
  ```

- Using unsigned integers

  To apply *forced nest-loop-join* and *making multiple SQL objects*, specify the following:

  ```
  pd_optimize_level=4,10
  ```

- When you have upgraded from HiRDB Version 5.0 or older version

  The total value specified in HiRDB Version 5.0 or older version is also valid. If there is no need to change the optimization option, you need not change the specification of this operand after upgrading to HiRDB Version 6 or a newer version. To add the optimization option, use the following example:

  Example:

  Forced nest-loop-join and *making multiple SQL objects* have been applied to HiRDB Version 5.0, and *priority of OR multiple index use* is to be newly added.

  ```
  pd_optimize_level = 14,128
  ```

  However, because this specification makes it difficult to identify the facilities being applied, use of identifier specification is recommended.

Operand rules

- Identifiers and unsigned integers cannot be specified together.
- Identifier specification

  - Enclose each SQL optimization option in quotation marks (").
  - If no SQL optimization option is used, specify `NONE`. When `NONE` and an identifier are both specified, the specification of `NONE` is invalid.

- Identifiers can be specified in uppercase or lowercase letters.

- Specifying the same identifier more than once is the same as specifying it once.

- Unsigned integer specification

  - If you do not use the SQL optimization option explained here, specify `0`. However, if both `0` and a non-zero unsigned integer are specified, the specification of `0` is ignored.

  - Specifying the same unsigned integer more than once is the same as specifying it once.

Specification guidelines

For specification guidelines, see *SQL optimization option* in the *HiRDB Version 8 UAP Development Guide*.

Notes

- If SQL optimization is specified in an SQL statement, SQL optimization specification takes precedence over the specification in this operand. For details on SQL optimization specification, see the manual *HiRDB Version 8 SQL Reference*.
- If SQL optimization is specified in an SQL statement (`CREATE PROCEDURE`, `CREATE TYPE`, `ALTER PROCEDURE`, `CREATE TRIGGER`, `ALTER ROUTINE`, or `ALTER TRIGGER`) inside a stored routine or trigger, the SQL optimization option inside the SQL statement takes precedence over the specification in this operand.

Relationship to other operands

- When the `pd_floatable_bes` or `pd_non_floatable_bes` operand is specified in the front-end server definition, specification of "increasing the target floatable servers (back-end servers for fetching data)" and "limiting the target floatable servers (back-end servers for fetching data)" is invalid.
- When `KEY` is specified for the `pd_indexlock_mode` operand, specification of "suppressing creation of update-SQL work tables" is invalid.

Relationship to client environment definition

The value of this operand can be different for each client. To change the value for a client, specify the `PDSQLOPTLVL` operand in the client environment definition. For details about the `PDSQLOPTLVL` operand, see the *HiRDB Version 8 UAP Development Guide.*

5) pd_additional_optimize_level = *SQL-extension-optimizing-option*

[,*SQL-extension-optimizing-option*]...

～ <identifier or unsigned integer>

Specifies SQL extension optimizing options. For details about the SQL extension optimizing options, see the *HiRDB Version 8 UAP Development Guide.*

The SQL extension optimizing option facilities are described below.

SQL extension optimizing option facility	Identifier	Unsigned integer
Application of optimizing mode 2 based on cost	`"COST_BASE_2"`	1
Hash execution of a hash join or a subquery[1]	`"APPLY_HASH_JOIN"`	2
Deterring of foreign server execution in SQL statement containing join[1, 2]	`"DETER_JOIN_SQL"`	67108864
Forcing of foreign server execution in SQL statement containing direct product[1, 2]	`"FORCE_CROSS_JOIN_SQL"`	134217728
Deterring of derivation of search acceleration condition that is generated unconditionally and that can be executed in foreign server[1, 2]	`"DETER_FSVR_DERIVATIVE_COND"`	1073741824

[1] These items are valid when `application of optimizing mode 2 based on cost` is specified.

[2] These items are valid for searching a foreign table. Otherwise, they are invalid.

Operand specification methods

Select the SQL extension optimization options to be applied and specify their identifiers or unsigned integers. Although either identifiers or unsigned integers (computed values) can be used to specify options, the use of identifiers is usually recommended.

- Using identifiers

  To apply "application of optimizing mode 2 based on cost" and "hash execution of a hash join or a subquery," specify the following:

  ```
  pd_additional_optimize_level= "COST_BASE_2",
  "APPLY_HASH_JOIN"
  ```

- Using unsigned integers

  To apply "application of optimizing mode 2 based on cost" and "hash-execution of a hash join or a subquery," specify the following:

  ```
  pd_additional_optimize_level=1,2
  ```

Operand rules

- Identifiers and unsigned integers cannot be specified together.
- Identifier specification
  - Enclose each SQL extension optimizing option in quotation marks (").
  - If you do not use the SQL extension optimizing option explained here, specify `NONE`. However, if both `NONE` and an identifier other than `NONE` are specified, the specification of `NONE` is invalid.
  - Identifiers can be specified in uppercase or lowercase letters.
  - Specifying the same identifier more than once is the same as specifying it once.
- Unsigned integer specification
  - If you do not use the SQL extension optimizing option explained here, specify `0`. However, if both `0` and a non-zero unsigned integer are specified, the specification of `0` is invalid.
  - Specifying the same unsigned integer more than once is the same as specifying it once.

Specification guidelines

For specification guidelines, see *SQL extension optimization option* in the *HiRDB Version 8 UAP Development Guide*.

Notes

- If the SQL optimization is specified in an SQL statement, SQL optimization specification takes precedence over the specification in this operand. For details on SQL optimization specification, see the *HiRDB Version 8 SQL Reference*.
- If the SQL extension optimizing option is specified in an SQL statement (`CREATE PROCEDURE`, `CREATE TYPE`, `CREATE TRIGGER`, `ALTER PROCEDURE`, `ALTER ROUTINE`, or `ALTER TRIGGER`) inside a stored routine or trigger, the SQL extension optimizing option inside the SQL statement takes precedence over the specification in this operand.

Relationship to client environment definition

The value of this operand can be different for each client. To change the value for a client, specify the `PDADDITIONALOPTLVL` operand in the client environment definition. For details about the `PDADDITIONALOPTLVL` operand, see the *HiRDB Version 8 UAP Development Guide.*

6) pd_floatable_bes = "*back-end-server-name*"[,"*back-end-server-name*"]...

~ <identifier>((1-8 characters))

Specifies back-end servers to be used as floating servers.

Not all of the specified back-end servers may actually be used as floating servers.

Normally, servers used for data extraction are not used as floating servers. However, this operand enables servers that are used for data extraction to also become floating server candidates.

Operand rules

- If an undefined back-end server name is specified, it is ignored.
- If all the specified back-end server names are undefined, this entire operand is ignored.

Relationship to other operands

- If the `pd_non_floatable_bes` operand is also specified, it is invalidated.
- If the `pd_floatable_bes` operand is specified, "increasing the target floatable servers (back-end servers for fetching data)" of the `pd_optimize_level` operand is not applied.
- If the `pd_floatable_bes` operand is specified, "limiting the target floatable servers (back-end servers for fetching data)" of the `pd_optimize_level` operand is not applied.

7) pd_non_floatable_bes = "*back-end-server-name*"[,"*back-end-server-name*"]...

~ <identifier>((1-8 characters))

Specifies back-end servers that are not to be used as floating servers.

If all back-end servers are specified, this operand is invalid.

Operand rule

If an undefined back-end server name is specified, it is ignored.

Relationship to other operands

- If the `pd_floatable_bes` operand is also specified, it is invalidated.
- If the `pd_non_floatable_bes` operand is specified, "increasing the target floatable servers (back-end servers for fetching data)" of the `pd_optimize_level` operand is not applied.
- If the `pd_non_floatable_bes` operand is specified, "limiting the target floatable servers (back-end servers for fetching data)" of the `pd_optimize_level` operand is not applied.

6.3.3 Operands related to system monitoring

8) pd_watch_pc_client_time = ***maximum-client-request-wait-time***

~ <unsigned integer>((0-65535)) (Seconds)

Specifies in seconds the maximum amount of time for a server to wait for the next request from a HiRDB client after the HiRDB server returns a response to a request from a Windows-compatible HiRDB client.

If no request comes from the HiRDB client within the specified amount of time, it will be assumed that an error occurred at the client and the connection between the server and the client will be terminated forcibly. No notice of disconnection is sent to the HiRDB client in such a case.

The time that is monitored is the period between CONNECT and DISCONNECT (i.e., the non-transaction status time), excluding the period between SQL execution startup and COMMIT or ROLLBACK.

Notes

- When 0 is specified, the server waits indefinitely for the next request from the HiRDB client.
- If a small value (e.g., up to 600) is specified for this operand, the HiRDB client may detect *server down* during SQL execution and may be terminated incorrectly.
- For a UNIX version of a HiRDB client (including a Linux version of a HiRDB client), time is not monitored regardless of the value specified for this operand. To monitor time for a UNIX version of a HiRDB client, specify the PDSWATCHTIME client environment definition of the HiRDB client.

Relationship to client environment definition

The value of this operand can be modified for each client. To do so, the PDSWATCHTIME operand must be specified in the client environment definition. For details about the PDSWATCHTIME operand, see the *HiRDB Version 8 UAP Development Guide*.

9) pd_spd_syncpoint_skip_limit = ***maximum-number-of-skipped-synchronization-point-dumps***

~ <unsigned integer>((0, 2-100000))

If an event such as an endless loop occurs in a UAP, acquisition of synchronization point dumps may be skipped. If several synchronization point dumps are not acquired (instead, they are skipped), there will be an increase in the number of overwrite-disabled system log files, which may result in a shortage of system log file capacity and ultimately in abnormal termination of the unit.

This operand specifies the maximum number of times synchronization point dumps can be skipped (number of skips per transaction). When the number of skipped synchronization point dumps reaches the specified operand value, the target transaction is cancelled forcibly and rolled back. This is called the *skipped effective synchronization point dump monitoring facility*.

If Real Time SAN Replication based on the log-only synchronous method is used, the skipped effective synchronization point dump monitoring facility cannot be used at the log application site regardless of this operand value.

For details on this facility, see the *HiRDB Version 8 System Operation Guide*.

Advantage

Specifying this operand provides a means of dealing with endless loops in UAPs.

Specification guideline

Normally, specify `0` for this operand. When `0` is specified, HiRDB computes the upper limit for the skip count. If the specification of `0` causes a problem, change the value of this operand. For the method of estimating the specification value, see the *HiRDB Version 8 System Operation Guide*.

Notes

- The monitoring provided for by the specification of this operand begins the first time a synchronization point dump is collected after HiRDB is started (including a restart).
- When this operand is specified, UAPs that are being executed in the no-log mode are also monitored. If the processing of a UAP being executed in the no-log mode is cancelled, the database cannot be automatically recovered, and thus RDAREAs are placed in the error shutdown state. Therefore, when specifying the upper limit, account also for the size of the system logs that are output from other transactions inside the server during the processing of UAPs that are executed in the no-log mode.
- The `pdload`, `pdmod`, `pdrorg`, `pdexp`, `pddbst`, `pdgetcst`, `pdrbal`, and `pdvrup` commands are not monitored by this facility.

6.3.4 Operands related to SQL runtime warning output facility

10) pd_cwaittime_wrn_pnt =
***output-condition-for-SQL-runtime-warning-information (% specification)* |**
output-condition-for-SQL-runtime-warning-information (time specification)

Specify this operand when using the SQL runtime warning output facility. You can specify this operand using one of the following two methods:

- Specifying a percentage
- Specifying a time duration

For details on the SQL runtime warning output facility, see the *HiRDB Version 8 System Operation Guide*.

***output-condition-for-SQL-runtime-warning-information (% specification)*: ~ <unsigned integer>((0-99)) or <unsigned decimal number>((0-99.999999)) (%)**

Specifies the condition for outputting SQL runtime warning information of the SQL runtime warning output facility as a percentage of the maximum client wait time (value of the `PDCWAITTIME` operand in the client environment definition). After SQL execution, HiRDB checks the SQL execution time. If this SQL execution time is determined to exceed the time specified by this operand, the following warning information is output. This is called the *SQL runtime warning output facility*.

- SQL runtime warning information file
- Warning message (`KFPA20009-W`)

Specification guidelines

Specify this operand as a percentage (%) of the value of the `PDCWAITTIME` operand. For example, if `100` (seconds) is specified for the `PDCWAITTIME` operand and `90` (%) is specified for the `pd_cwaittime_wrn_pnt` operand, HiRDB checks the SQL execution time after SQL execution. If the determined SQL execution time is 90 seconds or longer but less than 100 seconds, the warning information is output.

Example

```
PDCWAITTIME = 100
pd_cwaittime_wrn_pnt = 90
```

***output-condition-for-SQL-runtime-warning-information (time specification)*: ~ <unsigned decimal number>((0-65534.999999))sec (seconds)**

Specifies the condition for outputting SQL runtime warning information of the SQL runtime warning output facility as a time duration. After SQL execution, HiRDB checks the SQL execution time. If this SQL execution time is determined to exceed the time specified by this operand, the following warning information is output. This is called the *SQL runtime warning output facility*.

- SQL runtime warning information file
- Warning message (`KFPA20009-W`)

Operand specification method

Specify the time duration (in seconds) to be used as the output trigger. (You can specify up to the sixth decimal.) Add sec to the specified value.

Example

```
pd_cwaittime_wrn_pnt = 0.001sec
```

The following explanation is the same for both percentage and time duration specification.

Operand rule

If 0 or 0sec is specified in this operand, no warning information is output. (The SQL runtime warning output facility is not used.)

Relationship to client environment definition

You can change the value of this operand for each client. To change it for each client, specify the PDCWAITTIMEWRNPNT operand of the client environment definition. For details on the PDCWAITTIMEWRNPNT operand, see the *HiRDB Version 8 UAP Development Guide*.

Relationship to other operands

This operand is related to the following operands:

- pd_cwaittime_report_dir
- pd_cwaittime_report_size

6.3.5 Operand related to the facility for output of extended SQL error information

11) pd_uap_exerror_log_use = YES | NO

Specifies whether to use the facility for output of extended SQL error information. For details about this facility, see the *HiRDB Version 8 UAP Development Guide*.

YES:

Uses the facility for output of extended SQL error information. SQL error information is output in an error log file and an SQL error report file.

NO:

The facility for output of extended SQL error information is not used.

Relationship to client environment definition

The value of this operand can be changed for each client. To change the operand for a client, specify the PDUAPEXERLOGUSE operand in the client environment definition. If both this operand and the PDUAPEXERLOGUSE operand in the client environment definition are specified, the PDUAPEXERLOGUSE operand takes precedence.

For details about the PDUAPEXERLOGUSE operand, see the *HiRDB Version 8 UAP Development Guide*.

6.3.6 Operands related to lock

12) pd_fes_lck_pool_size = ***front-end-server-lock-pool-size***

~ **<unsigned integer> (KB)**

- 32-bit mode: **((1-2000000))**
- 64-bit mode: **((1-2000000000))**

Specifies in kilobytes the size of the shared memory area to be used for locking by a front-end server.

Specification guidelines

- In the 32-bit mode, the amount of lock pool space required for six lock requests is 1 KB.
- In the 64-bit mode, the amount of lock pool space required for four lock requests is 1 KB.
- Use the following formula to determine the value of this operand:

$\uparrow (a + b) \div c \uparrow$ (KB)

a: Total number of transaction lock requests to be executed concurrently by the front-end server. The number of lock requests depends on the SQL. For details about how to determine the total number of lock requests, see *E. Determining the Number of Locked Resources*.

b: (*value of* pd_max_users + pd_max_reflect_process_count + 3) × (*value of* pd_max_access_tables + 4)

c: Use 6 and 4 for 32- and 64-bit modes, respectively.

Tuning the specified value

See the usage rate for the locked resources management table (%OF USE LOCK TABLE) displayed for the front-end server in the statistical information on system operation by the statistics analysis utility. If the maximum usage rate equals or exceeds 80%, it is recommended that the operand's value be increased in preparation for future database expansion. If the maximum usage rate does not exceed 10%, it is recommended that the operand's value be decreased to conserve shared memory space.

Note

If the value specified for this operand is too small, an SQL may return an error.

13) pd_lck_hash_entry = *lock-pool-hash-entry-count*

～ <unsigned integer>((0-2147483647))

Specifies the number of hash table entries to be used in the lock pool. According to the value specified here, HiRDB allocates a lock pool in the shared memory for the unit controller.

Specification guidelines

Normally, omit this operand.

Consider specifying a value for this operand in the following cases:

- If you do not wish to change the shared memory size if possible when upgrading to Version 06-02 or a newer version, specify `11261`. In this case, the same number of hash entries is allocated as in the older version, and the hash table size inside the lock pool remains the same as before.
- Specifying for this operand a value greater than the recommended value may improve performance. However, specifying a value greater than variable *a* described in *Determining the recommended value* does not improve performance over a case in which *a* is specified.

Operand rules

- If `0` is specified for this operand or if this operand and the `pd_lck_hash_entry` operand of the server common definition are omitted, HiRDB computes a recommended value for each server. For details about recommended values, see *Specification guidelines* for the `pd_lck_hash_entry` operand.
- If a non-zero value or a non-prime number is specified for this operand, HiRDB assumes that the largest prime number not exceeding the specified value has been specified.

Note

If the value specified for this operand is too small, hash entry shortage may occur, resulting in performance degradation. If this operand is omitted, neither hash entry shortage nor performance degradation due to hash entry shortage occurs.

Determining the recommended value

For details on the recommended values, see *Determining the recommended value* for the `pd_lck_hash_entry` operand of the server common definition.

6.3.7 Operands related to buffers

14) pd_sql_object_cache_size = ***SQL-object-buffer-size***

~ **<unsigned integer> (KB)**

- 32-bit mode: **((22-256000))**
- 64-bit mode: **((22-2000000))**

Specifies in kilobytes the size of the buffer area (shared memory) in which SQL objects are to be placed.

Specification guidelines

- SQL objects are saved in a buffer until the user's transaction has terminated. The buffer must be large enough to store the SQL objects of all transactions that will be executed concurrently.
- SQL analysis can be reduced by saving the SQL objects of static SQLs in the buffer after transaction termination (until the buffer runs out of space) and sharing them among multiple users who execute the same UAP. To effectively utilize the buffer, it should be allocated so that the SQL objects of frequently-used UAPs are resident in the buffer.
- To estimate the buffer size, first determine the buffer size needed for UAP execution from the length of the SQL objects from the SQL statements to be issued by the UAP. Then, compute the buffer size by considering the number of UAPs that will be executed concurrently and the number of concurrently executing users.
- For details about how to estimate the length of the SQL object from a single SQL statement, see *D.2 Formulas for determining size of SQL object buffer (pd_sql_object_cache_size).*

Tuning the specified value

For details about how to tune the SQL object buffer size, see the *HiRDB Version 8 System Operation Guide*.

15) pd_table_def_cache_size = ***table-definition-information-buffer-size***

~ **<unsigned integer> (KB)**

- 32-bit mode: **((100-65535))**
- 64-bit mode: **((100-2000000))**

Specifies in kilobytes the size of the buffer (shared memory) for table definition information that has been used. Table definition information is used during pre-processing of an SQL statement.

Advantages

- Table definition information that has been used is kept in the memory as long as possible and thus can be used subsequently without an input operation.
- Performance improves when a large number of dynamic SQLs are used.
- The number of communications with the dictionary server is reduced.

Specification guidelines

- The total size of the definition information for frequently-used tables is specified.
- For details about how to determine the size of the table definition information buffer per table, see *D.3 Formulas for determining size of table definition information buffer (pd_table_def_cache_size)*.

Tuning the specified value

For details about how to tune the size of the table definition information buffer, see the *HiRDB Version 8 System Operation Guide*.

16) pd_auth_cache_size = *user-privilege-information-buffer-size*

~ <unsigned integer>((1-100)) (KB)

Specifies in kilobytes the size of the buffer (shared memory) for user privilege information.

Specification guidelines

- The user privilege information buffer stores CONNECT privilege, DBA privilege, and audit privilege information. If this buffer contains no information, information is obtained from a dictionary table during HiRDB connection, thus lengthening the response time. Therefore, specify a buffer size that can store the information for the users who are always connected.
- Storing the user privilege information of each user requires 68 bytes. Use this information when computing the total buffer size.

Tuning the specified value

For details about how to tune the size of the user privilege information buffer, see the *HiRDB Version 8 System Operation Guide*.

17) pd_view_def_cache_size = *view-analysis-information-buffer-size*

~ <unsigned integer> (KB)

- 32-bit mode: **((0-32000))**
- 64-bit mode: **((0-2000000))**

Specifies in kilobytes the size of the buffer (shared memory) for view analysis information.

Advantage

View analysis information that has been used is kept in the shared memory and can be used subsequently without an I/O operation.

Specification guideline

The total size of the view analysis information for frequently-used view tables is specified.

Tuning the specified value

For details about how to tune the size of the buffer for view analysis information, see the *HiRDB Version 8 System Operation Guide*.

18) pd_alias_cache_size = ***table-alias-definition-information-buffer-size***

~ <unsigned integer>((0-3000)) (KB)

Specifies in kilobytes the size of the buffer (shared memory) for table alias definition information.

Specification guidelines

This area should be large enough to hold the information for all frequently used table aliases. The size of the information on each table alias is approximately 164 bytes.

19) pd_type_def_cache_size = ***user-defined-type-information-buffer-size***

~ <unsigned integer> (KB)

- 32-bit mode: **((100-65536))**
- 64-bit mode: **((100-2000000))**

This operand is applicable to user-defined types; specification of this operand is recommended when user-defined types will be used.

Specifies in kilobytes the size of the buffer (shared memory) for information on user-defined types. When a user-defined type is used, information on it is stored in this buffer. This information is used during pre-processing of SQL statements.

Advantages

- When user-defined type information is stored in this buffer, it is not necessary to access the dictionary table when the same user-defined type is used subsequently, thus reducing the number of input operations and the CPU usage time.
- The number of communications with the dictionary server is reduced.

- Specification of this operand improves performance when many dynamic SQLs are used.

Specification guidelines

The specified size should be the total of the sizes of the definition information for frequently-used user-defined types that are defined in tables. The following formula can be used to determine the definition information size for one user-defined type:

$\uparrow\{((0.3 + 0.2 \times a + 0.1 \times b) + 3) \div 4\}\uparrow \times 4$ (KB)

a: Number of user-defined type attributes

b: Number of subtypes that have inherited a supertype

Tuning the specified value

For details about how to tune the size of the buffer for user-defined type information, see the *HiRDB Version 8 System Operation Guide*.

20) pd_routine_def_cache_size = *routine-definition-information-buffer-size*

~ **<unsigned integer> (KB)**

- 32-bit mode: **((0, 20-65536))**
- 64-bit mode: **((0, 20-2000000))**

Specifies in kilobytes the size of the buffer (shared memory) for storing the following types of definition information (this information is used during pre-processing of SQL statements):

- Plug-in facility definition information
- System definition scalar facility definition information
- Routine definition information

Advantages

- When these types of definition information are stored in this buffer, it is not necessary to access the dictionary table when the same information is used subsequently, thus reducing the number of I/O operations and CPU usage time.
- The number of communications with the dictionary server is reduced.

Application criteria

This operand should be specified when a large number of the following types of SQL statements are used:

- SQL statements that use a plug-in

- SQL statements that use the system definition scalar facility
- SQL statements that use a routine

Specification guidelines

For details about how to determine the value for this operand, see *D.5 Formulas for determining size of routine definition information buffer (pd_routine_def_cache_size)*.

Tuning the specified value

For details about how to tune the size of the routine definition information buffer, see the *HiRDB Version 8 System Operation Guide*.

Note

If the value specified for this operand is smaller than the total of the sizes of the definition information for all plug-ins, the definition information on plug-in facilities will not be allocated in the buffer.

21) pd_registry_cache_size = *registry-information-buffer-size*

~ <unsigned integer>((0-65536)) (KB)

This operand is related to plug-ins. If you use a plug-in that uses registry information, Hitachi recommends the use of this operand. Hitachi also recommends the use of this operand if you use HiRDB Text Search Plug-in.

Specify the size of the buffer (shared memory) for storing registry information (units: KB). When registry information is used, it is stored in the buffer. Registry information is used during the execution of an SQL statement.

Advantages

- Once registry information is stored in this buffer, it is not necessary to access the registry when the same information is used subsequently, thus reducing the number of I/O operations and CPU usage time.
- The number of communications with the dictionary server is reduced.
- Specifying this operand can improve performance when a plug-in that makes frequent use of registry information is used.

Specification guidelines

Use the following formula to determine the registry information buffer size:

$\uparrow(0.3 + a)\uparrow \times b$ (KB)

a: Average registry key length (KB)

The average registry key length can be determined with the following SQL statement:

```
SELECT AVG(KEY_LENGTH) FROM MASTER.SQL_REGISTRY
```

Because the result of this SQL statement is output in bytes, convert it to kilobytes.

b: Number of registry keys registered

The number of registry keys registered can be determined with the following SQL statement:

```
SELECT COUNT(*)FROM MASTER.SQL_REGISTRY
```

Tuning the specified value

For details about how to tune the size of the buffer for registry information, see the *HiRDB Version 8 System Operation Guide*.

6.3.8 Operands related to RPC trace information

22) pd_rpc_trace = Y | N

Specifies whether or not RPC trace information is to be collected. HiRDB maintenance information is output in the RPC trace information.

Normally, this operand should be omitted.

`Y`: Collect RPC trace information.

`N`: Do not collect RPC trace information.

Note

Specifying `Y` for this operand degrades communication performance.

23) pd_rpc_trace_name = "*name-of-RPC-trace-collection-file*"

~ <pathname of up to 254 characters>

Specifies an `absolute pathname` for the filename for the RPC trace files. Three RPC trace files are created, with 1, 2, and *l* suffixed to the specified file name.

Note

Files with a maximum size of `pd_rpc_trace_size` *value* × 2 are created under the directory specified by this operand. Attention should be paid to the file capacity.

24) pd_rpc_trace_size = *RPC-trace-collection-file-size*

~ <unsigned integer>((1024-2147483648)) (Bytes)

Specifies in bytes the size of the RPC trace files.

The value specified by this operand determines the size of both RPC trace file 1 and RPC trace file 2.

Specification guidelines

An RPC trace collects in a single file in time order the information on the communications among all processes. If the volume of this information exceeds the specified value, trace data output is switched to the other file, which means that older trace information will be overwritten. When this happens, the amount of trace information that is available may be inadequate, making troubleshooting difficult. For this reason, at least 1,000,000 should be specified for this operand.

Note

The value specified by this operand does not apply to RPC trace file *l*, because its size is fixed at 0 bytes.

6.3.9 Operands related to troubleshooting information

25) pd_module_trace_max = ***maximum-number-of-module-traces-that-can-be-stored***

～ <unsigned integer>((126-16383))

A HiRDB process records the history of the executed functions and macros inside the process private memory. This history is called a *module trace*. This operand specifies the number of module trace records. The content of this history is loaded into the `core` file and is output when a process error occurs.

Specification guidelines

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

Process private memory of the following size is allocated to each process:

In the 32-bit mode: 64 + 48 × *value of* `pd_module_trace_max` operand (bytes)

In the 64-bit mode: 64 + 64 × *value of* `pd_module_trace_max` operand (bytes)

26) pd_module_trace_timer_level = 0 | 10 | 20

Specifies how to acquire the time to be output in module traces. The following table explains the meaning of the value specified for this operand.

Specified value	Time acquisition method
`0`	Time is output in seconds at every module trace output location.
`10`	Time is output in microseconds only at performance-critical module trace output locations, such as those before and after input/output processing, and time is output in seconds at other locations.
`20`	Time is output in microseconds at every module trace output location.

Specification guidelines

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

If you specify a value other than `0` for this operand, a function for acquiring time in microseconds is issued, and as a result, system performance may decline.

6.3.10 Operands related to the security audit facility

27) pd_audit_def_buffer_size = *security-audit-information-buffer-length*

~ <unsigned integer>((1-2000000)) (kilobytes)

Specifies (in kilobytes) the buffer size (shared memory) used for storing information for the security audit facility.

Specification guidelines

Use the following formula to determine the security audit information buffer length:

$\uparrow 0.3 + a \times 0.25 \uparrow$ (kilobytes)

a: Number of objects for the narrowing condition that was specified in the audit trail of the security audit facility

Notes

If the amount of memory required by this operand's specification cannot be allocated, HiRDB will not start.

6.3.11 Operands related to distributed database

28) pd_node_name = local-node-RD-node-name

~ <identifier>((1-30 characters))

This operand is applicable only to the HP-UX and AIX 5L versions.

When HiRDB is used as a client, specifies the local node's RD node name so that local access will be recognized when the specified name is specified as the RD node name in an SQL statement.

Specification of this operand is applicable when the distributed database facility is used. Because this operand increases the amount of memory used by front-end server processes, it should not be specified except when the distributed database facility is used.

6.3.12 Operands related to Java

Java operands are specified when a Java stored procedure or Java stored function is used. For details about Java stored procedures and Java stored functions, see the *HiRDB Version 8 UAP Development Guide.*

Note that some platforms on which HiRDB is run do not support Java stored procedures and Java stored functions. For details, see *Environment supporting Java stored procedures and functions* in the *HiRDB Version 8 System Operation Guide*.

29) pd_java_stdout_file = *"Java-virtual-machine-standard-output-and-standard-error-output-destination-file"*

~ **<pathname>**

Specifies as an absolute pathname the file to which the standard output and standard error output are to be output in a Java virtual machine. If this operand is omitted, the standard output and standard error output of the Java virtual machine are ignored.

Specification guideline

Because the size of the file specified by this operand is extremely large, this operand is not normally specified. It is recommended that this operand be specified during debugging of a Java stored procedure or Java stored function. There is no limit to the size of the file that can be specified by this operand.

Note

If there are simultaneous writing attempts from multiple processes, their output contents cannot be guaranteed.

Operand rules

- Up to 255 characters can be used for the pathname.

6.3.13 Operands related to system log files

30) pd_log_dual = Y | N

Specifies whether or not dual system log files are to be used.

`Y`: Use dual system log files.

`N`: Do not use dual system log files.

Advantages

When dual system log files are used, HiRDB collects the same system log information in both files, which are called File A and File B. If a failure occurs in one of the files while the collected system log file is being loaded, the file can be loaded from the other file, resulting in higher system reliability.

Relationship to other operands

When use of dual system log files is specified, the name of the File B system log file must be specified with the `pdlogadpf` operand.

31) pd_log_dual_write_method = serial | parallel

This operand is applicable only to the AIX 5L version of HiRDB.

This operand specifies whether to use the facility for parallel output of system logs. For details about this facility, see the *HiRDB Version 8 Installation and Design Guide*.

`serial`: The facility for parallel output of system logs is not used.

`parallel`: The facility for parallel output of system logs is used.

When `parallel` is specified for this operand, HiRDB uses the Asynchronous I/O Subsystem to execute parallel output to the system log files.

Conditions

When you specify `parallel` for this operand, the following conditions must be satisfied:

- `pd_log_dual = Y`
- The system log files are located in character special files.

If either of these conditions is not satisfied, the facility for parallel output of system logs is not used regardless of the value specified for this operand.

Additionally, when you specify `parallel` for this operand, the Asynchronous I/O Subsystem must be installed.

32) pd_log_remain_space_check = warn | safe

Specifies the processing to be performed by HiRDB when the available space in the system log file falls below the warning level. This is called the facility for monitoring the free area for the system log file. For details on this facility, see the

HiRDB Version 8 System Operation Guide.

`warn`:

When the available space in the system log file falls below the warning level, the `KFPS01162-W` message is output.

`safe`:

When the available space in the system log file falls below the warning level, scheduling of new transactions is suppressed, all transactions inside the server are forcibly terminated, and the `KFPS01160-E` message is output.

Specification guideline

Hitachi recommends the specification of `safe` because it can reduce the probability of abnormal termination of units due to system log file space shortage. However, when `safe` is specified, all transactions inside the server are forcibly terminated when a shortage occurs in the available space in the system log file. Therefore, the design of the system log file requires more accuracy. For details on system log file design, see the *HiRDB Version 8 Installation and Design Guide*.

33) pd_log_auto_unload_path = "*unload-log-file-output-directory*"[,"*unload-log-file-output-directory*"]...

～ <pathname>((1-136 characters))

Specifies as absolute pathnames the unload file output directories when the automatic log unloading facility is to be used for the system log. A HiRDB file system area name must be specified to create the unload log file in a HiRDB file system area. *The directories or HiRDB file system areas specified for this operand must be created before HiRDB is started.*

Additionally, in this operand, specify a different directory or HiRDB file system area for each server.

For details about the automatic log unloading facility, see the *HiRDB Version 8 System Operation Guide*.

Specification guidelines

It is important to check the available disk space before specifying a directory, so as to ensure that the created unload log file does not cause a disk space shortage.

If an unload log file cannot be created in the specified directory because of a disk space shortage, the automatic log unloading facility stops. If this is a possibility, creation of multiple directories is recommended.

Note, however, that the database recovery operation of selecting the unload files needed for recovery is simplified somewhat when only one directory is used.

Also keep in mind the following when multiple directories are created:

- It is recommended that directories be specified in different partitions to protect against disk errors.
- If the unload log file cannot be created in a single directory because of a full disk or disk error, create an unload log file under a different directory. HiRDB uses the directories specified by this operand in the order of their specification.

Operand rules

- Up to 128 directories can be specified.
- When multiple directories are specified, the same pathname cannot be specified.

Notes

The automatic log unloading facility cannot be used in the following cases:

- `N` is specified in the `pd_log_unload_check` operand.
- The log application site uses Real Time SAN Replication based on the log-only synchronous method (the `KFPS04689-W` message is displayed).

34) pd_log_singleoperation = Y | N

This operand is applicable when dual system log files are used; it need not be specified when dual system log files are not used.

Specifies whether or not the single-operation mode is to be used for the system log files. Even if an error occurs in a system log file, making no dual system log files available, HiRDB (or a unit for a HiRDB/Parallel Server) can continue processing using the remaining single normal system log file without being abnormally terminated. This is called *single operation of the system log files*.

The mode in which continuation of processing is permitted only with both system log files available (normal operation mode) is called *double operation of the system log files*.

`Y`: Use single operation of the system log files.

`N`: Do not use single operation of the system log files. Both system log files must always be used.

Condition

This operand is valid only when `pd_log_dual = Y` is specified.

35) pd_log_rerun_reserved_file_open = Y | N

Specifies whether or not a system log file is to be opened automatically.

If no system log file that can be overwritten is available during a unit restart, HiRDB opens a reserved file (if one is available), makes it overwritable, and continues processing. This is called *automatic opening of system log file*.

A reserved file is used in the following cases:

- Between the time of a restart and the time when the first synchronization point dump is collected
- When none of the opened file groups can be overwritten

`Y`: Open a system log file automatically (open and use a reserved file).

`N`: Do not open a system log file automatically (do not use a reserved file).

Advantages

When `Y` is specified, the unit can be restarted as long as a reserved file is available, even though no file that can be swapped in is available during the unit restart.

However, if a file that is in unload wait status is available, the unit is stopped; once that file has been unloaded, the unit can be restarted.

36) pd_log_rerun_swap = Y | N

Specifies whether or not the system log files are to be swapped during a unit restart.

`Y`: Swap the system log files.

`N`: Do not swap the system log files.

Advantage

When `Y` is specified, there can be a physical separation between the system log files used before and after a restart. Therefore, the system log file that was being used before the restart can be reused during server operation.

37) pd_log_swap_timeout = *wait-time-for-completion-of-system-log-file-swapping*

~ <unsigned integer>((1-32580)) (seconds)

Specifies in seconds the wait time during which swapping of system log files must be completed. If a swap of system log files is not completed within the specified amount of time, the unit terminates abnormally.

Specification guidelines

Normally, there is no need to specify this operand. If it takes a long time to swap system log files for a reason such as poor machine performance, you can specify this operand with a value that is greater than the default value. To detect errors or delays quickly during system log file swapping so that the unit can be terminated (such as because of a disk failure), reduce the value of this operand.

38) pd_log_unload_check = Y | N

Specifies whether or not HiRDB is to check the unload status of system log files.

`Y`:

Check the unload status (normal operation).

`N`:

Do not check the unload status. A system log file is placed in swappable status, regardless of its unload status, when both of the following conditions are satisfied:

- It is in overwritable status
- It is in extraction completed status (HiRDB Datareplicator)
- Overwriting permitted status for online reorganization (HiRDB Staticizer Option)

In this case, the system log file operation method is to release checking of unload status. For details about this operation method, see the *HiRDB Version 8 System Operation Guide*.

Advantages

Specifying `N` provides the following advantages:

- Operations are simplified because the system log file unload operation is eliminated.
- It is not necessary to provide files for storing unload files.

Specification guideline

`N` should be specified if the system log file will not be needed for database recovery (i.e., if recovery from a backup collection point will be sufficient).

Notes

The following points apply when `N` is specified:

- Database can be recovered only if backups have been made.
- If this option is specified when the system log file is required for database recovery, it will not be possible to recover the database.

39) pd_log_max_data_size = *log-input/output-buffer-size*

~ <unsigned integer>((32000-523000)) (Bytes)

Specifies in bytes the size of the buffer to be used for system log input/output operations.

Specification guideline

Specify a value that satisfies conditional expression 1 below. If `uap` is specified in the `pd_rpl_reflect_mode` operand or a recovery-unnecessary front-end server is used, specify a value that satisfies both the conditional expressions below (1 and 2). To optimize the value, use the tuning method for the specification value.

Conditional expression 1: log input/output buffer length ≥ *a*

a: 72 x total number of back-end servers + 1432

Conditional expression 2: log input/output buffer length ≥ *b*

b: (Maximum number of back-end servers subject to update processing by a single transaction + 1) x 128 + 64

Tuning the specified value

A value (other than the default value) may need to be specified for this operand after the following types of statistics analysis utility statistical information related to system operation have been checked:

- Number of buffer sectors waiting for input/output (`# OF BUFFER FOR WAIT I/O`)

  If the average number of buffer sectors waiting for input/output significantly exceeds 100, increase the value for this operand so that the average approaches 100.

- Number of waits caused by lack of a current buffer (`# OF WAIT THREAD`)

  If the number of waits caused by lack of a current buffer is not 0, increase the value for this operand.

Relationship to other operands

Use this operand and the `pd_log_write_buff_count` operand to determine the log output buffer size.

40) pd_log_write_buff_count = *log-output-buffer-sectors-count*

～ <unsigned integer>((3-65000))

Specifies the number of buffer sectors to be used for system log output.

Tuning the specified value

A value (other than the default value) may need to be specified for this operand after the following types of statistics analysis utility statistical information related to system operation have been checked:

- Number of times the buffer was full (# OF BUFFER FOR WAIT I/O)
- Number of waits caused by lack of a current buffer (# OF WAIT THREAD)

If either of these values is large, a larger value should be specified in order to improve throughput.

Relationship to other operands

Use this operand and the pd_log_max_data_size operand to determine the log output buffer size.

41) pd_log_rec_leng = *system-log-file-record-length*

～ <unsigned integer>((1024, 2048, 4096)) (Bytes)

Specifies the record length for the system log files; the specifiable values are 1024, 2048, and 4096.

The record length specified in the -l option of the pdloginit command should be specified for this operand.

Notes

- If a value that is different from the record length specified by the -l option of the pdloginit command is specified for this operand, system log files cannot be opened.
- For details about how to modify the system log file record length, see the *HiRDB Version 8 System Operation Guide*.

6.3.14 Operands related to synchronization point dump files

42) pd_spd_dual = Y | N

Specifies whether to use dual synchronization point dump files.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for the server are also used at the log application site.

Y: Uses dual synchronization point dump files.

N: Does not use dual synchronization point dump files.

Advantage

When dual synchronization point dump files are used, HiRDB collects the same synchronization point dump in both dump files A and B. Even when an error occurs in one of the files when the collected synchronization point dump is being loaded, the synchronization point dump can be loaded from the other file, thus improving system reliability.

Relationship to other operands

To use dual synchronization point dump files, specify the name of synchronization point dump file B in the `pdlogadpf` operand.

43) pd_spd_assurance_msg = Y | N

Specifies whether or not the `KFPS02183-I` message is to be output when a synchronization point dump is completed.

`Y`: Output the message.

`N`: Do not output the message.

44) pd_spd_assurance_count = *number-of-guaranteed-valid-generations*

~ <unsigned integer>((1-2))

~ <unsigned integer>((1-2))

Specifies the range of system log files to be saved during HiRDB operation as a protection against events such as a synchronization point dump file input error during system recovery. What is specified here is a number of synchronization point dump file generations; the specified value is referred to as the *number of guaranteed-valid generations*. The number of past generations of synchronization point dump files specified here are overwrite disabled.

If Real Time SAN Replication based on the log-only synchronous method is used, this operand is ignored, if specified, at the log application site. The system log files used at the log application site are overwritten by the transaction execution site even if they include system log information that has not been applied.

Advantage

When `2` is specified as the number of guaranteed-valid generations and an error occurs in the most recent synchronization point dump file generation, the system can be recovered using the preceding synchronization point dump file generation, resulting in higher reliability.

Specification guideline

- To improve reliability, `2` should be specified for the number of guaranteed valid generations. However, the number of overwrite disabled synchronization point dump files increases (to two).
- If reliability has been improved with the use of dual synchronization point dump files, Hitachi recommends that you omit this operand or specify `1` for it.

Notes

- The minimum number of synchronization point dump files needed is *number-of-guaranteed-valid-generations* + 1.
- Specifying 2 increases the number of overwrite disabled synchronization point dump files. Moreover, the system log files that correspond to the overwrite disabled synchronization point dump files are also overwrite disabled. Consequently, specifying 2 for the number of guaranteed valid generations increases the number of overwrite disabled system log files. As a result, a shortage may occur in the number of system log files that can be swapped in. To prevent this, it may be necessary to reevaluate the system log file capacity.

45) pd_spd_reduced_mode = *reduced-mode-operation-option*

~ <unsigned integer>((0-2))

Specifies whether or not the reduced mode operation for synchronization point dump files is to be used.

Reduced mode operation is a facility for continuing processing if at least two synchronization point dump files can be used, even if an event such as a file error during HiRDB operation or restart reduces the number of synchronization point dump files to or below the number of required files (*number of guaranteed valid generations*[*] + 1).

[*] Value specified for the pd_spd_assurance_count operand.

0: Do not use the reduced mode operation.

1: Use the reduced mode operation.

2: Use the reduced mode operation and issue a warning message whenever a synchronization point dump is being acquired during the reduced mode operation.

46) pd_spd_reserved_file_auto_open = Y | N

Specifies whether or not a synchronization point dump file is to be opened automatically. When Y is specified and an error in a synchronization point dump file reduces the number of synchronization point dump files to the number of files necessary for operation (*number of guaranteed valid generations*[*] + 1), HiRDB opens a reserved file (if one is available), makes it overwritable, and continues processing. This is called *automatic opening of synchronization point dump file*.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for the server are also used at the log application site.

[*] Value specified for the pd_spd_assurance_count operand.

Y:

Open a synchronization point dump file automatically. A reserved file is opened when the number of files falls below the number necessary for operation (*number of guaranteed valid generations* + 1).

`N`:

Do not open a synchronization point dump file. No reserved file is opened, even though the number of files falls below the number necessary for operation (*number of guaranteed valid generations* + 1).

Relationship to other operands

This operand has a higher priority than the `pd_spd_reduced_mode` operand.

47) pd_spd_max_data_size = *synchronization-point-dump-file-buffer-size*

~ <unsigned integer>((32000-4000000)) (Bytes)

Specifies in bytes the size of the buffer (shared memory) to be used for synchronization point dump file input/output operations.

The value specified here affects the number of synchronization point dump file input/output operations.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for the server are also used at the log application site.

Specification guidelines

- Normally, this operand need not be specified.
- When the value of this operand is increased, the number of synchronization point dump file input/output operations is reduced.

48) pd_log_sdinterval = *system-log-output-volume*[,interval]

Specifies the collection interval for synchronization point dumps. The following should be taken into consideration in specifying this operand:

- Volume of system log information output since the previous synchronization point
- Amount of time that has elapsed since the previous synchronization point

***system-log-output-volume*: ~ <unsigned integer>((100-100000)) (Number of log blocks)**

Specifies an interval between synchronization point dumps in terms of the number of blocks of log information. A synchronization point dump is collected each time system log information equivalent to the number of log blocks specified here has been output.

***interval*: ~ <unsigned integer>((0 or 10-1440)) (Minutes)**

Specifies a synchronization point dump collection interval in terms of number of minutes between synchronization point dumps.

- When `0` is specified for the interval, HiRDB does not use a time interval for collecting synchronization point dumps.
- If no transactions execute during an interval, no synchronization point dump is collected even though the amount of the time specified here has elapsed.

Specification guidelines

- This operand need not be specified if no specific amount of time is specified for restarting HiRDB.
- The value specified for this operand affects the amount of time required to restart HiRDB.
- Specifying a small value for this operand reduces the amount of time required for database recovery during a HiRDB restart. However, because the frequency of synchronization point dumps increases, online performance may deteriorate in some cases.

  Conversely, specifying a large value for this operand increases the amount of time required for database recovery during a HiRDB restart. However, because the frequency of synchronization point dumps decreases, online performance may improve in some cases.

Tuning the specified value

The synchronization point dump collection interval can be checked with the statistics analysis utility; the relevant information is shown under `SYNC POINT GET INTERVAL` in the statistical information related to system operation. The average `SYNC POINT GET INTERVAL` value should be used. If the synchronization point dump collection interval is determined to be too long, the specification value should be decreased; conversely, if it is determined to be too short, the specification value should be increased.

Note

The synchronization point dump collection interval is determined on the basis of the volume of system log information that is output. Therefore, committing data from memory to a database takes a long time during intervals that have few updating transactions. If an error occurs at such a time, it will take longer to recover the transactions that were generated during that period. If this is a possibility, the synchronization point dump collection interval should be set by also using the `interval` value.

6.3.15 Operands related to server status files

49) pd_sts_file_name_1 = "*logical-file-name*", "*file-a-status-file-name*", "*file-b-status-file-name*"

: . . .

pd_sts_file_name_7 = "*logical-file-name*", "*file-a-status-file-name*", "*file-b-status-file-name*"

Define server status files. Although `pd_sts_file_name_2` to `7` operands may be omitted, the `pd_sts_file_name_1` operand cannot be omitted.

"*logical-file-name*" ～ <identifier>((1-8 characters))

Specifies the logical file name of a status file for the front-end server.

When a command that manipulates the status file is to be executed, the logical file name defined here must be specified.

"*file-a-status-file-name*" ～ <pathname>((up to 167 characters))

Specifies the name of the File A status file as an absolute pathname.

"*file-b-status-file-name*" ～ <pathname>((up to 167 characters))

Specifies the name of the File B status file as an absolute pathname.

Specification guidelines

- The files specified as File A and File B should be status files created with the `pdstsinit` command.
- If a file that has not been created with the `pdstsinit` command is specified, a virtual status file is created.
- If an error occurs in a status file, HiRDB swaps the status files. If no spare file is available for swapping, HiRDB (or a unit for a HiRDB/Parallel Server) terminates abnormally. For this reason, defining a large number of system files improves system reliability, but at the expense of increasing the amount of disk space that is required.
- The status files specified for File A and File B must have the same record length and capacity.

Operand rules

- Up to seven instances of this operand may be specified.
- A status file has a dual structure consisting of File A and File B; both must be specified.
- Environment variables cannot be used for the absolute pathnames of the File A and File B file names.

- The same names cannot be specified for the logical file name, the File A file name, and the File B file name.

Notes

- When HiRDB is started normally, the primary file (the file that was the primary file when HiRDB was terminated) is inherited. However, if there is no current file that can be inherited, for example because all status files have been initialized, the first status file that was specified among those specified by the `pd_sts_file_name_1` to `7` operands becomes the current file, and the remaining files become spare files if they can be opened. Any files that cannot be opened become reserved files.
- When HiRDB is restarted, the primary file (the file that was the primary file when HiRDB was terminated) is inherited.

Using a virtual status file

Specifying a virtual status file makes it possible to add a new status file while HiRDB is running.

For example, if the number of spare files is reduced because of errors in status files, virtual status files can be converted into spare files. The procedure for converting virtual status files into spare files follows.

Procedure

1. Use the `pdstsinit` command to create a status file in a HiRDB file system area for system files.
2. Use the `pdstsopen` command to open the status file.

These operations can be performed while HiRDB is running; there is no need to stop HiRDB.

- **Advantages and disadvantages**

  Defining a virtual status file reduces the amount of space required in the HiRDB file system area. However, a virtual status file cannot be added as a spare file if the HiRDB file system area for system files does not have sufficient space (sufficient space for adding the file), thus reducing system reliability.

  When virtual status files are not defined, the amount of space required in the HiRDB file system area is increased. However, because a swap destination is guaranteed when a file error occurs, system reliability is increased.

- **Notes**

  When virtual status files are defined, HiRDB determines that a server status file error has occurred when the server is started. For this reason, `pd_sts_initial_error = continue` must be specified. It is also necessary before starting the unit to specify in the `pd_sts_last_active_file` operand the current file that was active during the previous operation.

Relationship to other operands

This operand is related to the `pd_sts_subfile_name_1` to `7` operands.

50) pd_sts_subfile_name_1 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"

:

pd_sts_subfile_name_7 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"

Defines the server status files for log application processing, which are used at the log application site when Real Time SAN Replication based on the log-only synchronous method is used.

"*logical-file-name*": ~ <identifier> ((1-8 characters))

Specifies the logical file name of a server status file for log application processing. This must be a logical file name specified in `pd_sts_file_name_1` through `pd_sts_file_name_7`. A logical file name defined here is used for executing commands that manipulate the status files for log application processing.

"*primary-status-file-name-for-log-application-processing*": ~ <path name> ((up to 167 characters))

Specifies the absolute path name of a primary status file for log application processing.

"*secondary-status-file-name-for-log-application-processing*": ~ <path name> ((up to 167 characters))

Specifies the absolute path name of a secondary status file for log application processing.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.

Specification guidelines

- For the primary and secondary status file names for log application processing, specify the names of status files for log application processing that were created in the preparations for log application. If any other status file for log application processing is specified, the specified file becomes a *virtual status file for log application processing*.
- If an error occurs in a status file for log application processing, HiRDB swaps status files for log application processing. If no spare file is available for swapping, HiRDB (or the unit in the case of a HiRDB/ Parallel Server) terminates abnormally. For this reason, defining a large number of status files improves system reliability, but at the expense of increasing the amount of disk space that is required.
- The specified primary and secondary status files for log application processing must have the same record length and capacity.

Operand rules

- You can specify a maximum of 7 instances of this operand.
- This operand is ignored at the transaction execution site.
- If you performed the preparations for log application during HiRDB startup, create status files for log application processing that correspond to all status files for transaction processing that can be opened and that were specified in `pd_sts_file_name_1` through `pd_sts_file_name_7`.
- The system uses dual status files for log application processing. Make sure that you specify both files.
- Environment variables cannot be used for the absolute path names of the primary and secondary status files for log application processing.
- Each logical file name, primary status file name for log application processing, and secondary status file name for log application processing must be unique.
- The names of the primary and secondary status files for log application processing must be different from the primary and secondary status file names specified in the `pd_sts_file_name_1` through `pd_sts_file_name_7` operands.

Notes

When HiRDB is started normally or restarted, the current file (the file that was the current file when HiRDB last terminated) is inherited. However, if there is no current file that can be inherited, such as when all status files for log application processing have been initialized, the log application site can no longer be started. If this occurs, perform the preparations for log application.

How to use a virtual status file for log application processing

See the description of the `pd_sts_file_name_1` through `pd_sts_file_name_7` operands.

Relationship to other operands

This operand is related to the `pd_sts_file_name_1` through `pd_sts_file_name_7` operands.

6.3.16 Operands related to server status files (when an error occurs)

For details about the measures to be taken when an error occurs in a status file, see the *HiRDB Version 8 System Operation Guide*.

51) pd_sts_initial_error = stop | continue | excontinue

When a server (single server, front-end server, dictionary server, or back-end server) starts, HiRDB performs a process of identifying the current server status file. This operand specifies the action that HiRDB takes when any of the following errors is detected during this identification process.

- No real server status file is found.
- An error is detected in the server status file.

The current file identification process is applied to the server status files specified by the `pd_sts_file_name_1` to `7` operands.

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing specified in the `pd_sts_subfile_name1` through `pd_sts_subfile_name7` operands are also used at the log application site. For the log application site, replace the `pd_sts_last_active_file` and `pd_sts_last_active_side` operands in the above description with the `pd_sts_last_active_subfile` and `pd_sts_last_active_side_sub` operands, respectively.

`stop`:

When an error is detected in a server status file during the current file identification process, the startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started. In this case, first take a corrective action for the status file in which the error was detected, and then start the unit.

`continue` or `excontinue`:

Even when an error is detected in the server status file during the current file identification process, the startup of the server is continued if the current file is normal. However, the startup may be stopped depending on the value specified for the `pd_sts_singleoperation` operand (whether operation should continue with a single status file). The following table shows the relationship to the `pd_sts_singleoperation` operand.

- **Relationship to the pd_sts_singleoperation operand**

pd_sts_singleoperation operand value	Processing by HiRDB	HiRDB administrator's action
`continue`	When an error is detected in a server status file, HiRDB cannot identify the current file, and thus startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.	The HiRDB administrator identifies the current file and specifies the `pd_sts_last_active_file` and `pd_sts_last_active_side` operands. Afterwards, start HiRDB.
`stop` (default value)	When an error is detected in a server status file, HiRDB identifies the current file and startup of the server is continued. However, if the primary and secondary files satisfy any of the conditions listed in the table below (cases in which HiRDB cannot identify the current file), startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.	If HiRDB cannot identify the current file, the HiRDB administrator identifies the current file and specifies the `pd_sts_last_active_file` and `pd_sts_last_active_side` operands. Afterwards, start HiRDB.

- **When HiRDB cannot identify the current file**

pd_sts_initial_error operand value	File A status	File B status
`continue`	Error shutdown	Error shutdown
	Error shutdown	Open (initial state)
	Error shutdown	No real file
	Open (initial state)	Error shutdown
	Open (initial state)	No real file
	No real file	Open (initial state)
	No real file	Error shutdown
	No real file	No real file

pd_sts_initial_error operand value	File A status	File B status
`excontinue`	Error shutdown	Error shutdown
	Error shutdown	No real file
	No real file	Error shutdown
	No real file	No real file

Therefore, to minimize actions by the HiRDB administrator (to reduce the number of cases in which both the `pd_sts_last_active_file` and `pd_sts_last_active_side` operands must be specified), specify the following:

- `pd_sts_initial_error = excontinue`
- `pd_sts_singleoperation = stop`

Specification guidelines

The following table shows the specification guidelines, along with their advantages and disadvantages.

Item	pd_sts_initial_error operand value	
	stop	continue or excontinue
Processing by HiRDB during server startup	When an error is detected in a server status file, startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/ Parallel Server) is not started.	Even when an error is detected in a server status file, the startup of the server is continued if the current file is normal.
Specification guideline	To improve system reliability, specify `stop`.	To simplify the error-handling actions during HiRDB startup, specify `continue` or `excontinue`.
Advantage	Guarantees that all server status files of the server are normal when the server starts. Therefore, if an error occurs in the current file after HiRDB has started, it can be swapped to a spare file.	Even when an error is detected in a server status file during server startup, HiRDB can start with the remaining normal files only. Therefore, HiRDB stop time can be shortened. In this case, because the number of spare files has become small, it is necessary to immediately repair the status files containing errors.
Disadvantage	Possibility increases that an error in a server status file stops the startup of HiRDB.	Because HiRDB may be running with only a small number of spare files, system reliability is low. Depending on the number of spare files available, it may not be possible to swap server status files.

Notes

- If both current files are abnormal, startup of the server is stopped regardless of the value specified for this operand, and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.
- Before starting HiRDB, do not initialize the current file with the `pdstsinit` command. If the current file is initialized, HiRDB cannot be restarted.

Remarks

For details about the operand value, HiRDB processing, and HiRDB administrator's action, see *Remarks* in the section on the `pd_sts_initial_error` operand.

52) pd_sts_singleoperation = stop | continue

Specifies whether or not processing may continue in the single-operation mode for server status files. The status file single-operation mode means that processing is to continue using only the normal file (single file) when an error occurs in the status file and a spare file is not available. For details about the status file single-operation mode, see the *HiRDB Version 8 Installation and Design Guide*.

If an error occurs in one of the current files and a spare file is available, the status file is swapped and processing continues regardless of the value specified for this operand (operation in the single-operation mode does not occur).

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing are also used at the log application site.

`stop`:

Do not permit operation in the single-operation mode. If operation in the single-operation mode is necessary, HiRDB (or a unit for a HiRDB/Parallel Server) is abnormally terminated. If HiRDB is abnormally terminated, allocate spare files and then restart HiRDB.

`continue`:

Enable operation in the single-operation mode. When the single-operation mode goes into effect, the message `KFPS01044-I` is output. If an error occurs in the normal file during operation in the single-operation mode or if HiRDB (or a unit for a HiRDB/Parallel Server) is abnormally terminated while the status file is being updated, HiRDB cannot be restarted. Therefore, when the single-operation mode goes into effect, immediately allocate spare files.

Specification guidelines

- Hitachi recommends that you specify `stop` to increase system reliability. Hitachi also recommends that you increase the number of spare files to guard against errors in the current files.
- The following table shows the specification guidelines, along with their advantages and disadvantages.

Item	pd_sts_singleoperation operand value	
	stop	continue
Specification guideline	To improve system reliability, specify `stop`.	Specify `continue` if it is important not to stop HiRDB.
Advantage	When an error occurs in one of the current files and a spare file is not available, operation in the single-operation mode does not occur and HiRDB is abnormally terminated. Consequently, the possibility of losing the content of the current files is reduced.	Even when an error occurs in one of the current files and a spare file is not available, processing can be continued. Therefore, the possibility that an error in the status file stops HiRDB is reduced.
Disadvantage	Possibility that an error in the status file stops HiRDB increases. However, increasing the number of spare files can reduce this possibility.	If an error occurs in the normal status file during operation in the single-operation mode or if HiRDB is abnormally terminated during updating of the status file, the content of the current file is lost and HiRDB cannot be restarted.

Relationship to other operands

The combination of the values specified for the `pd_sts_singleoperation` and `pd_sts_initial_error` operands determines the processing by HiRDB when an error occurs in the status file. Therefore, the values to be specified for these operands should be determined together.

53) pd_sts_last_active_file = "*logical-file-name*"

~ <identifier> ((1-8 characters))

Specifies the name of the logical file to be used as the current status file at the time of the front-end server startup. HiRDB compares the file name specified in this operand with the file name selected by HiRDB to be the current file. If the file names match, HiRDB (or a unit for a HiRDB/Parallel Server) is started; otherwise, HiRDB (or a unit for a HiRDB/Parallel Server) is not started.

Conditions

The following conditions must be satisfied:

- continue or excontinue is specified in the pd_sts_initial_error operand.
- It cannot be determined if the current file selected by the HiRDB system was the most recent current file during the previous session.

Specification guidelines

1. To start HiRDB immediately after initializing all status files:

   From among the operable logical files specified in the pd_sts_file_name1 to 7 operands, specify the one that has the smallest number. Forced startup will be used in this case, regardless of the previous termination mode.

2. When both of the current files are normal:

   Specify the name of the current file.* If HiRDB cannot be started even though the name of the current file is specified, the current file may have been initialized. In this case, first initialize all status files, then use the method in 1 above to start HiRDB (forced startup will be used, regardless of the previous termination mode).

3. When one of the current status files has an error:

   Use the method in 2 previously, with the following operands specified:

   - pd_syssts_singleoperation = continue
   - pd_sts_last_active_side

4. When both of the current status files have errors:

   Initialize all status files, then execute the method in 1 above (forced startup will be used, regardless of the previous termination mode).

5. When a virtual status file is specified

   Specify the name of the current file.*

* The names of the current files (that were active at the end of the previous operation) can be determined from the following messages:

- KFPS01001-I
- KFPS01010-E
- KFPS01011-I
- KFPS01063-I

Of the status files displayed by these messages, the one that is reported in the message that was output most recently is the current file.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for transaction processing are used at the transaction execution site. At the log application site, this operand is ignored, if specified.

54) pd_sts_last_active_side = A | B

Specify this operand if you want to start the front-end server when one of the current files is in an error state. Specify the normal status file for this operand. HiRDB compares the file specified in this operand with the file selected by HiRDB. If they match, HiRDB copies the contents of the normal status file to secondary File A and File B, then HiRDB switches the spare file in as the current file and starts the unit. If the files do not match, HiRDB (or a unit for a HiRDB/Parallel Server) is not started.

Conditions

The following operands must be specified:

- `pd_sts_initial_error = continue` or `excontinue`
- `pd_sts_last_active_file`

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for transaction processing are used at the transaction execution site. At the log application site, this operand is ignored, if specified.

55) pd_sts_last_active_subfile = "*logical-file-name*"

~ <identifier> ((1-8 characters))

When Real Time SAN Replication based on the log-only synchronous method is used, this operand specifies the logical file name of the status file for log application processing that is to be used as the current file when HiRDB (or the unit in the case of a HiRDB/Parallel Server) starts. HiRDB compares the file specified in this operand with the current file selected by HiRDB. If the file names match, HiRDB starts; if not, HiRDB does not start. Note that this operand is ignored if HiRDB is started immediately after the database is inherited pursuant to execution of the `pdrisedbto` command.

Conditions

For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*

The following conditions must be satisfied:

- Y must be specified in the pd_rise_use operand, and also syssync must be specified in the pd_rise_pairvolume_combination operand.
- continue or excontinue is specified in the pd_sts_initial_error operand.
- It cannot be determined if the current file selected by the HiRDB system was the most recent current file during the previous session.

Specification guidelines

1. When starting HiRDB immediately after initializing all status files for log application processing

   Perform the preparations for log application. Once the preparations for log application have been completed successfully, start HiRDB without specifying this operand.

2. When both of the current files are normal

   Specify the name of the current file.# If HiRDB cannot be started even though the name of the current file is specified, the current file may have been initialized. In such a case, first initialize all status files for log application processing, then use the method in 1 above to start HiRDB. Note that in this case, forced startup will be used, regardless of the previous termination mode.

3. When one of the current files has an error

   Use the method in 2 above and specify the following operands:

   - Specify continue in the pd_syssts_singleoperation operand
   - Specify the pd_sts_last_active_side_sub operand

4. When both of the current files have errors

   Perform the preparations for log application. Once the preparations for log application have been completed successfully, start HiRDB without specifying this operand.

5. When a virtual status file for log application processing is specified

   Specify the name of the current file.#

#

The names of the current files (that were active at the end of the previous session) can be determined from the following messages:

- KFPS01001-I

- KFPS01010-E
- KFPS01011-I
- KFPS01063-I

Of the status files for log application processing that are displayed in these messages, the one that is reported in the message that was output most recently is the current file.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing are used at the log application site. At the transaction execution site, this operand is ignored, if specified.

56) pd_sts_last_active_side_sub = A | B

When Real Time SAN Replication based on the log-only synchronous method is used, specify this operand if you wish to start HiRDB (or the unit in the case of a HiRDB/Parallel Server) while one of the current files is in error status. In such a case, you specify the normal file in this operand. HiRDB compares the file specified in this operand with the file selected by HiRDB. If their names match, HiRDB copies the contents of the normal file into the secondary file. HiRDB then designates the secondary file as the current file and starts itself. If the file names do not match, HiRDB does not start. This operand is ignored if HiRDB is started immediately after the database was inherited pursuant to execution of the `pdrisedbto` command.

Conditions

For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*

The following conditions must be satisfied:

- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.
- `continue` or `excontinue` is specified in the `pd_sts_initial_error` operand.
- The `pd_sts_last_active_subfile` operand is specified.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing are used at the log application site. At the transaction execution site, this operand is ignored, if specified.

6.3.17 Operands related to system log file configuration

57) pdlogadfg -d sys -g *file-group-name* [ONL]

Specifies a file group for a system log file. This operand cannot be omitted and must be specified. Use the `pdlogadpf` operand to assign system log files to the file group specified here.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name* ~ <identifier>((1-8 characters))

Specifies a name for the file group. All file group names must be unique within a server.

`ONL`:

Specify this option to enable (open) this file group when HiRDB is running. You can specify 2 to 200 file groups with `ONL` specified.

Operand rule

This operand must be specified at least twice but no more than 200 times.

Notes

Before adding, modifying, or deleting this operand when HiRDB Datareplicator (extracted side) is linked, terminate the corresponding HiRDB Datareplicator. If this operand is modified while HiRDB Datareplicator is running, its extraction process may fail.

58) pdlogadpf -d sys -g *file-group-name* -a "*system-log-file-name*" [-b "*system-log-file-name*"]

Specifies the system log files that comprise the file group. This operand cannot be omitted; it must be specified once for each file group.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name* ~ <identifier>((1-8 characters))

Specifies the file group name specified by the `pdlogadfg` operand. The file group name must be unique within the unit.

-a "*system-log-file-name*" ~ <pathname>((up to 167 characters))

Specifies an absolute path name as the name of the system log file that comprises the file group. Specify the name of the system log file that was initialized using the `pdloginit` command. Note that the system log file name must be unique within the unit.

-b "*system-log-file-name*" ～ <pathname>((up to 167 characters))

Specifies an absolute path name as the name of the system log file B when dual system log files are to be used (`pd_log_dual = Y` specified). If `pd_log_dual = Y` is not specified, the system log file name is invalid, even if it is specified.

Specify the name of the system log file that was initialized using the `pdloginit` command. Note that the system log file name must be unique within the unit.

Notes

Before adding, modifying, or deleting this operand when HiRDB Datareplicator (extracted side) is linked, terminate the corresponding HiRDB Datareplicator. If this operand is modified while HiRDB Datareplicator is running, its extraction process may fail.

6.3.18 Operands related to synchronization point dump file configuration

59) pdlogadfg -d spd -g *file-group-name* [ONL]

Specifies a file group for synchronization point dump files. This operand cannot be omitted and must be specified. Use the `pdlogadpf` operand to assign synchronization point dump files to the file group specified here.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name* ～ <<identifier>((1-8 characters))

Specifies a name for the file group. All file group names must be unique within a server.

`ONL`:

Specify this option to enable (open) this file group when HiRDB is running. You can specify 2 to 30 file groups with `ONL` specified.

Operand rule

This operand must be specified at least twice but no more than 60 times.

60) pdlogadpf -d spd -g *file-group-name* -a "*synchronization-point-dump-file-name*" [-b "*synchronization-point-dump-file-name*"]

Specifies the synchronization point dump files that comprise the file group. This operand cannot be omitted; it must be specified once for each file group.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name*: ～ <identifier>((1-8 characters))

Specifies the file group name specified by the `pdlogadfg` operand. The file group name must be unique within the unit.

-a "*synchronization-point-dump-file-name*": ～ <pathname>((up to 167 characters))

Specifies an absolute path name as the name of the synchronization point dump file that comprises the file group. Specify the name of the synchronization point dump file that was initialized using the `pdloginit` command. Note that the synchronization point dump file name must be unique within the unit.

-b "*synchronization-point-dump-file-name*": ～ <pathname>((up to 167 characters))

Specifies an absolute path name as the name of the synchronization point dump file B when dual synchronization point dump files are to be used (`pd_spd_dual = Y` specified). If `pd_spd_dual = Y` is not specified, the synchronization point dump file name is invalid, even if it is specified.

Specify the name of the synchronization point dump file that was initialized using the `pdloginit` command. Note that the synchronization point dump file name must be unique within the unit.

61) pdlogadfg -d ssp-g *file-group-name*[ONL]

When Real Time SAN Replication based on the log-only synchronous method is used, this operand specifies a group of synchronization point dump files for log application processing that is to be used at the log application site. The `pdlogadpf` operand is used to allocate the synchronization point dump files for log application processing.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name*: ～ <identifier> ((1-8 characters))

Specifies the name of a file group. This file group name must correspond to a set of synchronization point dump files for transaction processing.

`ONL`:

Specifies that this file group is to be made available (opened) when HiRDB is running. You can specify 2 to 30 file groups with `ONL` specified.

If Real Time SAN Replication based on the log-only synchronous method is used, when HiRDB is started for the first time at the log application site after the preparations for log application have been performed, the file groups that need to be inherited from the transaction execution site become available when the single server starts, regardless of the specification of `ONL`.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.

Operand rules

You can specify 2 to 60 instances of this operand.

Notes

Specify a `pdlogadfg -d ssp` operand for each file group specified in a `pdlogadfg -d spd` operand.

62) pdlogadpf -d ssp -g *file-group-name* -a "*synchronization-point-dump-file-name-for-log-application-processing*" [-b "*synchronization-point-dump-file-name-for-log-application-processing*"]

Specifies synchronization point dump files for log application processing that constitute a file group. You must specify one `pdlogadpf -d ssp` operand for each file group.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name*: ~ <identifier> ((1-8 characters))

Specifies the name of a file group specified in a `pdlogadfg` operand. This file group name must correspond to a set of primary synchronization point dump files for transaction processing.

-a "*synchronization-point-dump-file-name-for-log-application-processing*": ~ <path name> ((up to 167 characters))

Specifies the absolute path name of a synchronization point dump file for log application processing that is to be included in the file group. This must be the name of a synchronization point dump file for log application processing that was created during the preparations for log application. Note that each synchronization point dump file name for log application processing must be unique within the same unit.

-b "*synchronization-point-dump-file-name-for-log-application-processing*": ~ <path name> ((up to 167 characters))

When dual synchronization point dump files for log application processing are used (`Y` is specified in the `pd_spd_dual` operand), this option specifies the absolute path name of a secondary synchronization point dump file for log application processing. If `Y` is not specified in the `pd_spd_dual` operand, the `-b` option is ignored, if specified. You must specify in this option the name of a synchronization point dump file for log application processing created during the preparations for log application. Note that each synchronization point dump file for log application processing must be unique within the same unit.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.

Notes

Specify a `pdlogadpf -d ssp` operand for each file group specified in a `pdlogadfg -d ssp` operand.

6.3.19 Operands related to Plug-ins

63) pdplgprm -n *plug-in-name* [-s *shared-memory-size*]

Specifies the name of a plug-in and the size of the memory to be shared by the plug-in. This operand should be omitted if no plug-ins are to be used or no plug-ins will run in front-end servers.

Conditions

The plug-in specified here must have been registered in HiRDB with the `pdplgrgst` command.

-n *plug-in-name*: ～ <identifier>((1-30 characters))

For details about the names of plug-ins that can be specified here, see the manuals for the plug-ins.

-s *shared-memory-size*: ～ <unsigned integer>((1-2000000)) <<0>> (KB)

Specifies in kilobytes the size of the shared memory to be used by the plug-in. For details about the size of the shared memory to be used by the plug-in, see the manual for the applicable plug-in.

6.3.20 Operands related to HiRDB External Data Access facility

For details on the HiRDB External Data Access facility, see the *HiRDB External Data Access Version 7 Description and User's Guide.*

Note that some platforms on which HiRDB is run do not support the HiRDB External Data Access facility. For details, see *HiRDB External Data Access* in the manual *HiRDB Version 8 Description*.

64) pdhubopt

Specifies the foreign server to which the Hub optimization information definition is applied. Specify this operand in one of the following formats:

- `pdhubopt -s` *foreign-server-name* `-f` *Hub-optimization-information-definition-file-name*
- `pdhubopt -d` *foreign-server-type* [`-v` *foreign-server-version*] `-f` *Hub-optimization-information-definition-file-name*

`-s` *foreign-server-name* ~ **<identifier consisting of 1-30 characters>**

Specifies the name of the foreign server specified by `CREATE SERVER`. Specify this option to perform Hub optimization for each foreign server. When you specify this option, you cannot specify the `-d` or `-v` option.

If the foreign server name specified in the `-s` option is not registered, the following operand specification is assumed if a *foreign-server*.opt file exists under `$PDCONFPATH`:

`pdhubopt -s` *foreign-server-name* `-f` *foreign-server-name*`.opt`

`-d` *foreign-server-type* [`-v` *foreign-server-version*]

Specify these options to perform Hub optimization for each foreign server type and each foreign server version. For each foreign server type and each foreign server version, specify the values indicated in the following table. If you specify these options, you cannot specify the `-s` option.

Foreign server type	-d option specification value[1]	-v option specification value[2]
HiRDB Version 6 or later	HIRDB	6.0
HiRDB Version 5.0		5.0
XDM/RD E2	XDMRD	6.0
Oracle8i	ORACLE	8.1.5
Oracle9i		9.2
Oracle 10g		10.1
DB2	DB2_UDB_OS390	6.0

[1] Same as the server type in the `TYPE` clause of `CREATE SERVER`.

[2] Same as the server version in the VERSION clause of CREATE SERVER.

- -f *Hub-optimization-information-definition-file-name*

Specifies the name of the Hub optimization information definition file. Specify only the file name, not the path name. Store this file under $PDDIR/conf.

Condition

HiRDB External Data Access is required.

Operand specification methods

- This operand can be specified more than once.
- To manage the Hub optimization information definition file of a specific foreign server separately, specify the -s (*foreign-server-name*) option.
- If you use multiple foreign servers and want to share the Hub optimization information definition file in the same DBMS, specify the -d (*foreign-server-type*) and -v (*foreign-server-version*) options.

Notes

Note the following when there are multiple Hub optimization information definition files: All Hub optimization information definition files that satisfy the conditions specified here are applied. The priority order is: -s specification, -d and -v specification, and -d only specification, in that order.

Example:

HiRDB Version 6 (server name: hirdb_v6_01) and HiRDB Version 5.0 are used as foreign servers. The pdhubopt operand is specified as follows:

```
pdhubopt -d HIRDB -f hubopt1          1
pdhubopt -d HIRDB -v 6.0 -f hubopt2   2
pdhubopt -s hirdb_v6_01 -f hubopt3    3
```

- Hub optimization information definition files 1 through 3 are applied to HiRDB Version 6. If same operands are specified, the specification value in *3* has the highest priority. If the same operand is specified in *1* and *2*, the specification value in *2* has a higher priority.
- Hub optimization information definition file 1 is applied to HiRDB Version 5.0.

Chapter

7. Dictionary Server Definition

This chapter explains the operands of the dictionary server definition.

This chapter contains the following sections:

7.1 Operand formats

A dictionary server definition defines information for a dictionary server. This section explains the formats used to specify the operands of a dictionary server definition. Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *7.2 Operands whose specification values can be changed* and *7.3 Operand explanations*.

For users who are creating HiRDB system definitions for the first time

The first step is to determine the values to be specified for the operands shown in boldface type. In principle, HiRDB can be started once the boldface operands have been specified.

No.	Format	Operand category
1	[set pd_max_dic_process = *maximum-number-of-activated-processes-per-dictionary-server*]*	Processes
2	[set pd_process_count = *resident-processes-count*[,*resident-processes-count-at-server-startup*]]*	
3	[set pd_server_cleanup_interval = *interval-for-stopping-nonresident-server-processes*]*	
4	[set pd_svr_castoff_size = *maximum-memory-size-used-by-server-process*]*	
5	[set pd_max_ard_process = *asynchronous-READ-process-count*]*	
6	[set pd_dfw_awt_process = *number-of-processes-to-be-written-in-parallel-in-deferred-write*]	
7	[set pd_work_buff_mode = each | pool]*	Work tables
8	[set pd_work_buff_size = *work-table-buffer-size*]*	
9	[set pd_work_buff_expand_limit = *work-table-buffer-expansion-limit*]*	
10	[set pd_spd_syncpoint_skip_limit = *maximum-number-of-skipped-synchronization-point-dumps*]*	System monitoring

No.	Format	Operand category
11	[set pd_dfw_syncpoint_skip_limit = *maximum-number-of-skipped-synchronization-point-dumps-resulting-from-delay-of-synchronization-point-dump-acquisition-due-to-deferred-write-processing*]*	
12	[set pd_lck_pool_size = *server-lock-pool-size*]*	Lock
13	[set pd_lck_until_disconnect_cnt = *total-number-of-tables-and-RDAREAs-to-be-locked-per-server-UNTIL-DISCONNECT-specification*]*	
14	[set pd_max_open_holdable_cursors = *maximum-number-of-holdable-cursors-that-can-be-concurrently-open-when-LOCK-statement-with-UNTIL-DISCONNECT-specification-is-not-executed*]	
15	[set pd_lck_hash_entry = *lock-pool-hash-entry-count*]*	
16	[set pd_dbsync_lck_release_count = *global-buffer-lock-release-interval-during-synchronization-point-processing*]*	
17	[set pd_sql_object_cache_size = *SQL-object-buffer-size*]*	Buffers
18	[set pd_dic_shmpool_size = *dictionary-server-shared-memory-size*]*	Shared memory
19	[set pd_rpc_trace = Y | N]*	RPC trace information
20	[set pd_rpc_trace_name = "*name-for-RPC-trace-collection-files*"]*	
21	[set pd_rpc_trace_size = *RPC-trace-collection-file-size*]*	
22	[set pd_module_trace_max = *maximum-number-of-module-traces-that-can-be-stored*]*	Troubleshooting information
23	[set pd_module_trace_timer_level = *module-trace-output-time-acquisition-method*]	
24	[set pd_max_add_dbbuff_no = *maximum-global-buffers-count-for-dynamic-addition*]*	Global buffer
25	[set pd_max_add_dbbuff_shm_no = *maximum-shared-memory-segments-count-for-dynamic-addition*]*	

No.	Format	Operand category
26	[set pd_java_stdout_file = "*Java-virtual-machine-standard-output-or-standard-error-output-destination-file*"]*	Java
27	**[set pd_log_dual = Y | N]***	System log files
28	[set pd_log_dual_write_method = serial | parallel]	
29	[set pd_log_remain_space_check = warn | safe]*	
30	[set pd_log_auto_unload_path = "*unload-log-file-output-directory*" [,"*unload-log-file-output-directory*"]...]	
31	[set pd_log_singleoperation = Y | N]*	
32	[set pd_log_rerun_reserved_file_open = Y | N]*	
33	[set pd_log_rerun_swap = Y | N]*	
34	[set pd_log_swap_timeout = *wait-time-for-completion-of-system-log-file-swapping*]*	
35	[set pd_log_unload_check = Y | N]*	
36	[set pd_log_max_data_size = *log-input/output-buffer-size*]*	
37	[set pd_log_write_buff_count = *log-output-buffer-sectors-count*]*	
38	[set pd_log_rec_leng = *system-log-file-record-length*]*	
39	**[set pd_spd_dual = Y | N]***	Synchronization point dump files
40	[set pd_spd_assurance_msg = Y | N]*	
41	[set pd_spd_assurance_count = *number-of-guaranteed-valid-generations*]*	

No.	Format	Operand category
42	[set pd_spd_reduced_mode = *reduced-mode-operation-option*]*	
43	[set pd_spd_reserved_file_auto_open = Y | N]*	
44	[set pd_spd_max_data_size = *synchronization-point-dump-file-buffer-size*]*	
45	[set pd_log_sdinterval = *system-log-output-volume*[,*interval*]]*	
46	**set pd_sts_file_name_1 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"**	Server status files
	[set pd_sts_file_name_2 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_3 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_4 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_5 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_6 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	[set pd_sts_file_name_7 = "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
47	[set pd_sts_subfile_name_1 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_2 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_3 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_4 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	

No.	Format	Operand category
	[set pd_sts_subfile_name_5 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_6 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_7 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
48	[set pd_sts_initial_error = stop | continue | excontinue]*	Server status files (when an error occurs)
49	[set pd_sts_singleoperation = stop | continue]*	
50	[set pd_sts_last_active_file = "*logical-file-name*"]	
51	[set pd_sts_last_active_side = A | B]	
52	[set pd_sts_last_active_subfile = "*logical-file-name*"]	
53	[set pd_sts_last_active_side_sub = A | B]	
54	**pdwork -v "*HiRDB-file-system-area-name*"[,"*HiRDB-file-system-area-name*"]...**	Work table files
55	**{{pdlogadfg -d sys -g *file-group-name* [ONL]}}**	System log file configuration
56	**{{pdlogadpf -d sys -g *file-group-name* -a "*system-log-file-name*" [-b "*system-log-file-name*"]}}**	
57	**{{pdlogadfg -d spd -g *file-group-name* [ONL]}}**	Synchronization point dump file configuration
58	**{{pdlogadpf -d spd -g *file-group-name* -a "*synchronization-point-dump-file-name*" [-b "*synchronization-point-dump-file-name*"]}}**	

No.	Format	Operand category
59	`{{{set pdlogadfg -d ssp -g` *file-group-name*`[ONL]}}`	
60	`{{set pdlogadpf -d ssp -g file-group-name -a` `"`*synchronization-point-dump-file-name-for-log-application-processing*`" [-b` `"`*synchronization-point-dump-file-name-for-log-application-processing*`"]}}`	
61	`{{[pdplgprm -n` *plug-in-name* `[-s` *shared-memory-size*`]]}}`	Plug-ins

* If this operand is omitted, the value specified in the same operand in the server common definition is used.

7.2 Operands whose specification values can be changed

The values of some of the dictionary server definition operands can be changed in the system common definition, unit control information definition, and server common definition. These operands are indicated below. After a planned termination, forced termination, or abnormal termination of HiRDB, some HiRDB system definition operands can be modified while others cannot be modified. The operands that can be modified are indicated below.

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
1	pd_max_dic_process	—	—	Y	—	Y	—	N	Y^2
2	pd_process_count	—	—	Y	Y	Y	Y	Y^2	Y^2
3	pd_server_cleanup_interval	—	—	Y	Y	Y	Y	Y	Y
4	pd_svr_castoff_size	—	—	Y	Y	Y	Y	Y	Y
5	pd_max_ard_process	—	—	Y	—	Y	Y	Y	Y
6	pd_dfw_awt_process	—	—	Y	—	Y	Y	Y	Y
7	pd_work_buff_mode	—	—	Y	—	Y	Y	Y	Y
8	pd_work_buff_size	—	—	Y	—	Y	Y	Y	Y
9	pd_work_buff_expand_limit	—	—	Y	—	Y	Y	Y	Y
10	pd_spd_syncpoint_skip_limit	—	—	Y	Y	Y	Y	Y	Y
11	pd_dfw_syncpoint_skip_limit	—	—	Y	—	Y	Y	Y	Y

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
12	pd_lck_pool_size	—	—	Y	—	Y	Y	Y[2]	Y
13	pd_lck_until_disconnect_cnt	—	—	Y	—	Y	Y	N	Y
14	pd_max_open_holdable_cursors	—	—	Y	—	Y	Y	N	N
15	pd_lck_hash_entry	—	—	Y	Y	Y	Y	Y	Y
16	pd_dbsync_lck_release_count	—	—	Y	—	Y	Y	Y	Y
17	pd_sql_object_cache_size	Y	—	Y	Y	Y	Y	N	Y
18	pd_dic_shmpool_size	—	—	Y	—	Y	—	Y[2]	Y[2]
19	pd_rpc_trace	Y	Y	Y	Y	Y	Y	Y	Y
20	pd_rpc_trace_name	Y	Y	Y	Y	Y	Y	Y	Y
21	pd_rpc_trace_size	Y	Y	Y	Y	Y	Y	Y	Y
22	pd_module_trace_max	Y	Y	Y	Y	Y	Y	Y	Y
23	pd_module_trace_timer_level	Y	Y	Y	Y	Y	Y	Y	Y
24	pd_max_add_dbbuff_no	—	—	Y	—	Y	Y	N	Y
25	pd_max_add_dbbuff_shm_no	—	—	Y	—	Y	Y	N	Y

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
26	`pd_java_stdout_file`	Y	Y	Y	Y	Y	Y	Y	Y
27	`pd_log_dual`	—	—	Y	Y	Y	Y	N	N
28	`pd_log_dual_write_method`	—	—	Y	Y	Y	Y	Y	Y
29	`pd_log_remain_space_check`	—	—	Y	Y	Y	Y	Y	Y
30	`pd_log_auto_unload_path`	—	—	—	Y	Y	Y	N	N
31	`pd_log_singleoperation`	—	—	Y	Y	Y	Y	N	N
32	`pd_log_rerun_reserved_file_open`	—	—	Y	Y	Y	Y	Y	Y
33	`pd_log_rerun_swap`	—	—	Y	Y	Y	Y	Y	Y
34	`pd_log_swap_timeout`	—	—	Y	Y	Y	Y	Y	Y
35	`pd_log_unload_check`	—	—	Y	Y	Y	Y	Y	Y
36	`pd_log_max_data_size`	—	—	Y	Y	Y	Y	Y[2,3]	Y[2]
37	`pd_log_write_buff_count`	—	—	Y	Y	Y	Y	Y[3]	Y
38	`pd_log_rec_leng`	—	—	Y	Y	Y	Y	Y	Y
39	`pd_spd_dual`	—	—	Y	Y	Y	Y	N	N

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
40	pd_spd_assurance_msg	—	—	Y	Y	Y	Y	Y	Y
41	pd_spd_assurance_count	—	—	Y	Y	Y	Y	N	N
42	pd_spd_reduced_mode	—	—	Y	Y	Y	Y	N	Y
43	pd_spd_reserved_file_auto_open	—	—	Y	Y	Y	Y	N	Y
44	pd_spd_max_data_size	—	—	Y	Y	Y	Y	N	N
45	pd_log_sdinterval	—	—	Y	Y	Y	Y	Y	Y
46	pd_sts_file_name_1-7	—	—	—	Y	Y	Y	Y[1]	Y[1]
47	pd_sts_subfile_name_1-7	—	—	—	Y	Y	Y	N	N
48	pd_sts_initial_error	—	—	Y	Y	Y	Y	Y	Y
49	pd_sts_singleoperation	—	—	Y	Y	Y	Y	Y	Y
50	pd_sts_last_active_file	—	—	—	Y	Y	Y	Y	Y
51	pd_sts_last_active_side	—	—	—	Y	Y	Y	Y	Y
52	pd_sts_last_active_subfile	—	—	—	Y	Y	Y	Y	Y
53	pd_sts_last_active_side_sub	—	—	—	Y	Y	Y	Y	Y

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
54	`pdwork`	—	—	—	—	Y	Y	N	N
55	`pdlogadfg -d sys`	—	—	—	Y	Y	Y	Y[1]	Y[1]
56	`pdlogadpf -d sys`	—	—	—	Y	Y	Y	Y[1]	Y[1]
57	`pdlogadfg -d spd`	—	—	—	Y	Y	Y	Y[1]	Y[1]
58	`pdlogadpf -d spd`	—	—	—	Y	Y	Y	Y[1]	Y[1]
59	`pdlogadfg -d ssp`	—	—	—	Y	Y	Y	Y[1]	Y[1]
60	`pdlogadpf -d ssp`	—	—	—	Y	Y	Y	Y[1]	Y[1]
61	`pdplgprm`	—	—	—	Y	Y	Y	Y	Y

Y: Yes, specification value can be modified.

N: No, specification value cannot be modified.

— : Specification value cannot be modified because operand is not applicable.

SYS: System common definition

UNT: Unit control information definition

SVR: Server common definition

FES: Front-end server definition

DS: Dictionary server definition

BES: Back-end server definition

[1] Operand that can be added only; it cannot be deleted or modified.

[2] If the specified value is too small, it may not be possible to restart HiRDB. When this is the case, restore the value to what is was before the change and then restart HiRDB.

[3] If the specified value is too large, it may not be possible to restart HiRDB. When this

is the case, restore the value to what is was before the change and then restart HiRDB.

7.3 Operand explanations

7.3.1 Operands related to processes

1) pd_max_dic_process = ***maximum-number-of-activated-processes-per-dictionary-server***

~ <unsigned integer>((1-2048))

Specifies the maximum number of processes that can be activated per dictionary server. When multiple front-end servers are used, processes that exceed the value specified in `pd_max_users` can sometimes become concentrated in a single dictionary server. Even when multiple front-end servers are not used, processes that exceed the value specified in `pd_max_users` can become concentrated in a single dictionary server if the number of executions of operation commands related to RDAREAs and global buffers (`pdbufls`, `pddbls`, `pdopen`, `pdclose`, `pdhold`, and `pdrels`) exceeds the value specified in `pd_max_users`.

The `pd_max_dic_process` operand specifies the maximum number of processes that can be activated per dictionary server when that number exceeds the value of the `pd_max_users` operand.

Specification guidelines

- The value determined by the following formula indicates the maximum number of processes that may become concentrated in a single dictionary server.

  `pd_max_users` value × *number of front-end servers*

  Specify a value for the `pd_max_dic_process` operand by using the value determined here as the upper limit and taking the degree of process concentration in a single back-end server into consideration. Specifying an unnecessarily large value may cause memory shortage.

- If the execution of operation commands related to RDAREAs or global buffers causes processes to become concentrated in the dictionary server, the number of operation commands to be concurrently executed is used as the maximum value.

- If the value specified is smaller than the `pd_max_users` value, the `pd_max_users` value is assumed as the default and a warning message (`KFPS01888-W`) is output.

- If updatable online reorganization is used, specify an operand value that satisfies the following condition. Otherwise, HiRDB cannot be started.

  `pd_max_dic_process` + `pd_max_reflect_process_count` ≤

2048

Tuning the specified value

For details about how to tune the maximum number of processes that can be activated, see the *HiRDB Version 8 System Operation Guide*.

2) pd_process_count = ***resident-processes-count*****[,*****resident-processes-count-at-server-startup*****]**

～ <unsigned integer>((0-2048))

resident-processes-count

Specifies the number of processes that can be made resident in the dictionary server. A resident process is a process that is activated at the time the server is started.

Advantage

By activating the processes used by transactions that can be processed concurrently by the dictionary server at the time of system startup and keeping them resident, the process startup time can be reduced even when new transactions are entered. However, HiRDB startup will take longer.

Specification guidelines

See *Specification guidelines* for the `pd_process_count` operand of the server common definition.

Tuning the specified value

For details about how to tune the resident processes count, see the *HiRDB Version 8 System Operation Guide*.

Notes

- Because the resident processes count has an effect on memory space availability, specifying an unnecessarily large number may prevent HiRDB from starting.
- If more processes than the resident process count are needed, additional processes are dynamically started, up to the maximum processes count allowed. However, depending on the value specified for the `pd_max_server_process` operand, it may not be possible to start all of the processes indicated by the maximum processes count.

Operand rule

See *Operand rules* for the `pd_process_count` operand of the server common definition.

resident-processes-count-at-server-startup

Specifies the number of processes that can be made resident during HiRDB startup.

It is common for resident processes to be activated during HiRDB startup. If there are many resident processes, the amount of time required for HiRDB startup increases proportionately. For example, it takes approximately 1 second to activate a process on a server machine with a 100-MIPS performance rating.

Following are the differences in processing that result depending on whether or not a resident processes count at server startup is specified:

- When there is no specification of a resident processes count at server startup (when, for example, `pd_process_count = 500` is specified)

  All 500 resident processes are activated during HiRDB startup, and HiRDB will not start until they are all activated. In this example, it would take a 100-MIPS server machine about 500 seconds to activate all the resident processes during HiRDB startup.

- When a resident processes count at server startup is specified (when, for example, `pd_process_count = 500, 50` is specified)

  Some of the resident processes (50 in this case) are activated during HiRDB startup, and the remaining resident processes are activated after HiRDB startup. HiRDB can be started as long as the specified number of resident processes is activated. In this example, it would take a 100-MIPS server machine about 50 seconds to activate 50 resident processes during HiRDB startup. The remaining 450 resident processes (in this example) would be activated after HiRDB startup.

Advantage

The amount of time required for HiRDB startup is reduced. You should use this option when you want to reduce the HiRDB startup time as much as possible, such as when you are using the system switchover facility.

Specification guideline

The specified value should be equal to the number of processes that will be required immediately after HiRDB startup is completed.

Notes

When you specify a resident processes count at server startup, you should recheck the value in the `PDCWAITTIME` operand of the client environment definition.

If more UAPs than the resident processes count at server startup are to be executed immediately following HiRDB startup, transaction processing may not be performed until after the remaining resident processes have been

activated. Therefore, if the value specified in the PDCWAITTIME operand of the client environment definition is small, it may not be possible to process some UAPs due to timeouts. For details about the PDCWAITTIME operand, see the *HiRDB Version 8 UAP Development Guide*.

3) pd_server_cleanup_interval = *interval-for-stopping-nonresident-server-processes*

~ <unsigned integer> ((0-1440)) (minutes)

Specifies in minutes the interval for checking for nonresident server processes in HiRDB that are to be stopped. The facility for stopping nonresident server processes is applied when the number of executing server processes exceeds the number of processes that can be made resident (value specified by the pd_process_count operand). The number of server processes that the facility stops is computed automatically by HiRDB.

Advantage

This facility improves the utilization rates of the memory and of process resources because it increases the number of nonresident processes that can be reused when the workload (number of server processes that are executing) peaks.

Specification guidelines

- For example, if there is only one hour a day during which peak workload occurs and the intervals between peaks within that hour are approximately two minutes apart, 2 should be specified for this operand.
- This facility does not have any effect if the number of server processes that execute concurrently during peak periods is always fewer than the number of resident processes. In this case, this operand should be omitted.

Tuning the specified value

Collect statistical information on system operations for each server for one week. Determine the workload peaks from the number of server processes being serviced (# OF PROCESSES ON SERVICE). If that peak value exceeds the currently set number of processes that can be made resident (value specified by the pd_process_count operand), determine the interval between individual peaks and set that number of minutes.

However, if there are ample resources, such as memory and CPU, in the server machine, adding the shortfall in the number of processes to the number of resident processes (i.e., increasing the value of the pd_process_count operand) would be more effective in improving performance than specifying the pd_server_cleanup_interval

operand.

Note

When this operand is omitted or `0` is specified, the system checks every 10 seconds for nonresident server processes that are waiting to be serviced and stops any such processes that are found.

4) pd_svr_castoff_size = *maximum-memory-size-used-by-server-process*

~ <unsigned integer>((0-2048)) (MB)

This operand need not be specified for the Linux version.

Specifies the maximum size of memory used by each server process processed by the dictionary server. If at the applicable trigger point shown below the amount of memory being used by a server process exceeds the value specified here, that server process is terminated. When a server process is so terminated, the `KFPS00350-w` message is output. This is called the *facility for monitoring the memory size of server processes*; for details, see the *HiRDB Version 8 System Operation Guide*.

Server type	Process name	Process termination trigger
Dictionary server	pddic	At transaction decision

Advantages

The facility for monitoring the memory size of server processes resolves the following problem:

- *The memory size of a server-resident process becomes too large during a particular SQL processing, significantly reducing the amount of system memory that is available.*

The HiRDB server releases memory space that is no longer needed. However, even when a program releases memory, the OS holds the memory area itself in the memory management facility inside the applicable process. Consequently, a process that becomes large in terms of using a large memory area even once never shrinks and continues to have an adverse impact on the system, especially in the case of resident processes. The facility for monitoring the memory size of server processes can prevent memory shortages because it terminates even resident processes.

Application criterion

Apply this facility when the amount of memory space used by a HiRDB server process becomes large, resulting in memory shortages.

Specification guidelines

- Maximum value for this operand

  Normally specify a value by considering the maximum processing capability of HiRDB. Assuming that the maximum number of SQL statements may be concurrently connected, determine for this operand a value that satisfies the following condition in each unit:

  $a \times (b + c) < d$

  a: Number of server processes within the unit (*maximum-number-of-concurrent-connections* × *server-count-inside-unit*)

  b: Virtual memory size for one server process immediately following HiRDB startup

  c: `pd_svr_castoff_size` operand value

  d: Memory size that can be allocated to a unit (memory size excluding the area used by other programs)

- Minimum value for this operand

  If the value specified for this operand is smaller than the memory size needed for normal SQL processing, the efficiency of making processes resident will deteriorate, resulting in frequent process terminations and restarts. Each time this occurs, a message is output to the syslogfile or the message log file, resulting in further performance degradation. To prevent this, select for this operand a value that satisfies the following condition for each server:

  $a - b < c$

  a: Server process virtual memory size following SQL process termination

  b: Virtual memory size for server process immediately following HiRDB startup

  c: `pd_svr_castoff_size` operand value

The virtual memory size can be determined by an OS command (e.g., the `top` command in HP-UX).

Operand rule

When `0` is specified for this operand, the facility for monitoring the memory size of server processes is not applied.

5) pd_max_ard_process = ***asynchronous-READ-process-count***

~ <unsigned integer>((0-256))

Specify this operand if you use the asynchronous READ facility. For this operand, specify the number of processes necessary for asynchronous READ operations. For a HiRDB/Parallel Server, this operand specifies the number of processes per server (back-end server or dictionary server). For details on the asynchronous READ facility, see the *HiRDB Version 8 Installation and Design Guide*.

Condition

A value of `1` or greater must be specified for the `-m` option of the `pdbuffer` operand.

Advantage

The asynchronous READ facility is especially effective (improves performance) when character special files, for which input and output take a long time, are used. Conversely, when regular files, for which input and output do not take a long time, or SANRISE disks are used, the asynchronous READ facility may not have much effect because of the following reasons:

- Input/out time does not overlap with CPU time most of the time.
- Communication processing has a large overhead.

Specification guidelines

- Specify `0` or `1`. However, if a value between 2 and 256 is specified for the `-m` option of the `pdbuffer` operand, specify the same value as the `-m` option value. If a value greater than 256 is specified for the `-m` option, specify the same value as the number of disk devices that store RDAREAs and system files (the number of such disk devices per server for a HiRDB/Parallel Server) or `256`.
- Increasing the value of this operand can shorten the processing time when the degree of concurrency is high for the SQL statements to which the asynchronous READ facility is applied. Decreasing the value of this operand may lengthen the processing time when the degree of concurrency is high for the SQL statements to which the asynchronous READ facility is applied. This is because asynchronous READ processes may have to wait for processing completion.
- Because a number of processes equaling *value-of-this-operand* × *server-count* are started, determine a value for this operand by taking resources (shared memory and message queue) into consideration. For the method of estimating shared memory and message queue sizes, see the *HiRDB Version 8 Installation and Design Guide*.

Tuning the specified value

For the method of tuning the specification value (the number of asynchronous READ processes), see the *HiRDB Version 8 System Operation*

Guide.

Relationship to other operands

If you change the value of this operand, reevaluate the value of the `pd_max_server_process` operand.

Operand rule

If you specify `0` for this operand, the asynchronous READ facility is not used.

6) pd_dfw_awt_process = *number-of-processes-to-be-written-in-parallel-in-deferred-write*

~ <unsigned integer>((2-255))

Specify this operand when you use the *facility for parallel writes in deferred write processing* for all buffer pools. Specify for this operand the number of processes to be processed in parallel. Increasing the number of processes can shorten the write processing time. For details about the facility *for parallel writes in deferred write processing*, see the *HiRDB Version 8 Installation and Design Guide*.

Specification guidelines

Specify `2`, which is the smallest value that enables the facility for parallel writes in deferred write processing. Furthermore, to determine the value for this operand, see *Tuning deferred write processing* in the *HiRDB Version 8 Installation and Design Guide*.

Note

Specifying the facility for parallel writes in deferred write processing increases the number of processes, and consequently raises the CPU usage rate.

7.3.2 Operands related to work tables

7) pd_work_buff_mode = each | pool

Specifies the method of allocating buffers when HiRDB creates tables.

`each`: Allocate a buffer for each work table.

`pool`: Allocate a buffer pool for each server process.

Specification guidelines

- Normally, `pool` is specified. `pool` is the appropriate specification when a large volume of data is to be retrieved and when manipulations such as `join`, `ORDER BY`, and `GROUP BY` are to be performed.
- When the size of the process private area that can be used for work table

buffers is predetermined, `pool` should be specified. When `pool` is specified, HiRDB efficiently allocates work table buffers to work tables.

In such a case, the process private area is occupied on the basis of the value specified in `pd_work_buff_size`, and input/output operations on work tables are buffered in that pool. Therefore, the process private memory is occupied only to the extent of the value specified in `pd_work_buff_size`.

8) pd_work_buff_size = ***work-table-buffer-size***

~ <unsigned integer> (KB)

- 32-bit mode: **((128-1000000))**
- 64-bit mode: **((128-4000000000))**

Specifies in kilobytes the size of buffers for work tables to be created by HiRDB.

Item	pd_work_buff_mode=each is specified	pd_work_buff_mode=pool is specified
Advantage	A large work table buffer size reduces the number of I/O operations associated with data manipulation, which means that the execution time of SQLs that use work tables is also reduced. However, because each server's process private memory is used, specification of this option should take into account the overall size of the system memory (real memory and virtual memory). If `pd_work_buff_mode` = each is specified, the memory size to be allocated is *value of* `pd_work_buff_size` × *required number of work tables*. A process private memory that has been allocated will remain allocated to that process at least in the virtual memory (will not be released until that process disappears) even after release is requested by HiRDB. Therefore, specifying an unnecessarily large value may cause a virtual memory shortage for other processes.	
Application criterion	`pd_work_buff_mode = pool` should be specified when a large volume of data is to be retrieved and when manipulations such as `join`, `ORDER BY`, and `GROUP BY` are to be performed.	

Item	pd_work_buff_mode=each is specified	pd_work_buff_mode=pool is specified
Specification guidelines	• Specify the size of the buffer to be allocated for one work table. • If a value greater than the work table memory capacity is specified for the work table buffer size, input/output to the work table file during work table creation is eliminated, thus reducing the time necessary for work tables. The following formula can be used to determine the work table memory capacity: `Work table memory capacity = Applicable work table size* ÷ 2`	• Specify the size of the buffer pool to be allocated for the entire server process. • Specify a value between 4352 and 5120 when a large volume of data is to be retrieved or when manipulations such as join, `ORDER BY`, and `GROUP BY` are to be performed. Specifying such a value increases the unit of sorting input/output, thus reducing the sort time. • If a large value is specified for the work table buffer size, with the total work table memory capacity for each SQL statement as the upper limit, input/output operations on the work table file during work table creation is eliminated, thus reducing the time necessary for work tables. The following formula can be used to determine the total work table memory capacity for each SQL statement: `Total work table memory capacity per SQL statement =` $a \times b + c \times d$
Notes	When multiple users execute processes concurrently or when an SQL statement that uses multiple work tables is executed, the specified size buffer is allocated for each work table. Consequently, specifying a large value may result in a memory shortage.	If the value specified for the work table buffer size is smaller than the number of work tables to be used by each SQL statement, the processing time may become longer than when `each` is specified. Specifically, specify a value that is at least equal to *maximum number of work tables for each SQL statement* × *128*. The following formula can be used to determine the maximum number of work tables for each SQL statement: `Maximum number of work tables for each SQL statement =` $b + d$
Operand rule	Specify a multiple of 128. If the value is not a multiple of 128, it is rounded up to the next multiple of 128.	• Specify a multiple of 128. If the value is not a multiple of 128, it is rounded up to the next multiple of 128. • Specify at least 384. If a value that is smaller than 384 is specified, it is rounded up to 384.

a:

↑ {Capacity of work table (for storing column information)* (KB) ÷ 2} ÷

128↑ × 128

b:

Maximum number of work tables (for storing column information)*

c:

↑{Capacity of work table (for storing positional information)* (KB) ÷ 2} ÷ 128↑ × 128

d:

Maximum number of work tables (for storing positional information)*

* For details about how to determine these values, see the *HiRDB Version 8 Installation and Design Guide*.

9) pd_work_buff_expand_limit = *work-table-buffer-expansion-limit*

∼ <unsigned integer> (KB)

- 32-bit mode: **((128-1000000))**
- 64-bit mode: **((128-4000000000))**

The size of the work table buffer to be created by HiRDB is specified by the `pd_work_buff_size` operand. Specify the `pd_work_buff_expand_limit` operand if you want to automatically expand a work table buffer when the space in this buffer becomes insufficient. The work table buffer is expanded up to the size specified by this operand.

For example, when the following values are specified for the operands, a 1,024-KB work table buffer is normally allocated. When this size becomes insufficient, the work table buffer is expanded up to 2,048 KB.

- `pd_work_buff_size = 1024`
- `pd_work_buff_expand_limit = 2048`

HiRDB expands a work table buffer in the following cases:

- The necessary work table buffer cannot be allocated when hash execution is applied to an execution method that uses hash join or subquery hash as the joining method.
- A 128-KB work table buffer allocated to each work table becomes insufficient when multiple work tables are concurrently used.

Condition

The `pd_work_buff_mode` operand must be omitted or `pool` must be specified for it.

Advantage

You can prevent a work table buffer shortage (too small a value specified for the pd_work_buff_size operand) from causing UAP errors.

Notes

- A work table buffer is not expanded when either of the following conditions is satisfied:
  - The pd_work_buff_expand_limit operand is not specified.
  - pd_work_buff_expand_limit operand value ≤ pd_work_buff_size operand value
- When a work table buffer is expanded, resident processes are also terminated at the following triggers (except in the Linux version) to free up memory:

  If a holdable cursor is not being used, a dictionary server process is terminated when a transaction is normally terminated or cancelled.

  If a holdable cursor is being used, a dictionary server process is terminated when a UAP is disconnected from HiRDB.

Operand rule

Specify a multiple of 128. If a value other than a multiple of 128 is specified, it is automatically rounded up to a multiple of 128.

Relationship to other operands

When a work table buffer is expanded for the first time in a dictionary server process, the KFPH29008-I message is output. Note that you can use the pd_work_table_option operand to suppress this message output.

Note

After a work table buffer has been expanded, when the number of work tables being used by the applicable server process goes to zero, the expanded work table buffer is released. The number of work tables being used can go to zero in the following cases:

- All cursors that were being used are closed. (In this case, the number of work tables being used may not go to zero.)
- A transaction is normally terminated or cancelled when a holdable cursor is not being used.
- A UAP is disconnected from HiRDB when a holdable cursor is being used.

Remarks

Hash join, subquery hash execution is applied in the following cases:

- *Application of optimizing mode 2 based on cost* and *hash join, subquery hash execution* are specified in the `pd_additional_optimize_level` operand, the `PDADDITIONALOPTLVL` operand of the client environment definition, or the `ADD OPTIMIZE LEVEL` operand of the SQL compile option.
- `HASH` is specified for the SQL optimization specification of the joining method inside an SQL statement.
- `HASH` is specified for the SQL optimization specification of the subquery execution method inside an SQL statement.

7.3.3 Operands related to system monitoring

10) pd_spd_syncpoint_skip_limit = ***maximum-number-of-skipped-synchronization-point-dumps***

~ <unsigned integer>((0, 2-100000))

If an event such as an endless loop occurs in a UAP, acquisition of synchronization point dumps may be skipped. If several synchronization point dumps are not acquired (instead, they are skipped), there will be an increase in the number of overwrite-disabled system log files, which may result in a shortage of system log file capacity and ultimately in abnormal termination of the unit.

This operand specifies the maximum number of times synchronization point dumps can be skipped (number of skips per transaction). When the number of skipped synchronization point dumps reaches the specified operand value, the target transaction is cancelled forcibly and rolled back. This is called the *skipped effective synchronization point dump monitoring facility*.

If Real Time SAN Replication based on the log-only synchronous method is used, the skipped effective synchronization point dump monitoring facility cannot be used at the log application site regardless of this operand value.

For details on this facility, see the *HiRDB Version 8 System Operation Guide*.

Advantage

Specifying this operand provides a means of dealing with endless loops in UAPs.

Specification guideline

Normally, specify `0` for this operand. When `0` is specified, HiRDB computes the upper limit for the skip count. If the specification of `0` causes a problem, change the value of this operand. For the method of estimating the specification value, see the *HiRDB Version 8 System Operation Guide*.

Notes

- The monitoring provided for by the specification of this operand begins the first time a synchronization point dump is collected after HiRDB is started (including a restart).
- When this operand is specified, UAPs that are being executed in the no-log mode are also monitored. If the processing of a UAP being executed in the no-log mode is cancelled, the database cannot be automatically recovered, and thus RDAREAs are placed in the error shutdown state. Therefore, when specifying the upper limit, account also for the size of the system logs that are output from other transactions inside the server during the processing of UAPs that are executed in the no-log mode.
- The `pdload`, `pdmod`, `pdrorg`, `pdexp`, `pddbst`, `pdgetcst`, `pdrbal`, and `pdvrup` commands are not monitored by this facility.

11) pd_dfw_syncpoint_skip_limit = *maximum-number-of-skipped-synchronization-point-dumps-resulting-from-delay-of-synchronization-point-dump-acquisition-due-to-deferred-write-processing*

～ <unsigned integer>((0-100000))

If a synchronization point occurs within the synchronization point dump acquisition interval but before deferred write processing is completed, acquisition of the synchronization point dump is skipped. This is because acquisition of the synchronization point dump is delayed by the deferred write processing and the number of update buffers output by the synchronization point exceeds the number of update buffers that can be output within the synchronization point dump acquisition interval.

If more than one synchronization point dump is skipped, there will be an increase in the number of system log files that cannot be overwritten, which may result in a shortage of system log file capacity and ultimately in abnormal termination of the unit.

This operand specifies the maximum number of times synchronization point dumps can be skipped (number of skips per transaction) due to deferred write processing.

When the number of skipped synchronization point dumps due to deferred write processing reaches the specified operand value, HiRDB determines the maximum number of update buffers in such a manner that acquisition of a synchronization point dump can be completed within the synchronization point dump acquisition interval. If the maximum number of update buffers is exceeded, HiRDB then outputs the oldest update buffer and limits the total size of update buffers at a synchronization point. This is called the *update buffer size restriction facility*.

Advantage

If a synchronization point occurs within the synchronization point dump acquisition interval but before deferred write processing has terminated, the unit terminates abnormally. This operand enables such abnormal termination of the unit to be avoided.

Specification guidelines

Normally, you will omit this operand. If you wish to prevent unit abnormal termination caused by the occurrence of a synchronization point before termination of deferred write processing within the synchronization point dump acquisition interval, specify 1.

If an acceptable number of times synchronization point dumps can be skipped can be determined in advance, such as from the size of the log information, specify that value.

Operand rules

If 0 is specified in this operand, HiRDB does not use the update buffer size restriction facility.

Notes

The following notes explain the period of effectiveness of the update buffer size restriction facility. The period of effectiveness means the interval between issuance of the `KFPH23035-I` message and issuance of the `KFPH23036-I` message.

- If the number of update buffers exceeds the maximum number of update buffers determined by HiRDB, update buffers are output after update processing has executed; this degrades the update processing throughput. You can use the following formula to obtain the maximum number of update buffers determined by HiRDB:

(synchronization point dump interval / unit value of `WRITE`#)
x (1 - (amount of log information from the previous synchronization point dump to the pre-synchronization point)
x (number of buffer sectors in buffer pool / total number of buffer sectors in buffer pool that was updated at the synchronization point)

  #: For details about the unit value of `WRITE`, see the *HiRDB Version 8 System Operation Guide*.

- If the deferred write trigger is specified in the `pd_dbbuff_rate_updpage` or `pdbuffer -y` operand, and each operand value becomes greater than the maximum number of update buffers determined by HiRDB, the maximum number of update buffers determined by HiRDB is changed to the number of update buffers that triggers deferred write processing.

The value of the `pdbuffer -w` operand is adjusted automatically so that up to the maximum number of update buffers is output for each buffer.

- A skipped synchronization point dump is detected during update buffer output processing at a synchronization point. Therefore, the update buffer size restriction facility may be enabled after a synchronization point dump is skipped and an error message is displayed.
- Normally, there is one output request per synchronization point for the number of processes that used the facility for parallel writes in deferred write processing during synchronization point processing (`pd_dfw_awt_process`).

  However, if the update buffer size restriction facility is used, there will be more than one output request in order to facilitate early detection of skipped synchronization point dumps.

7.3.4 Operands related to lock

12) pd_lck_pool_size = ***server-lock-pool-size***

～ <unsigned integer> (KB)

- 32-bit mode: **((1-2000000))**
- 64-bit mode: **((1-2000000000))**

Specifies in kilobytes the size of the shared memory to be used by the dictionary server for locking.

Specification guidelines

- In the 32-bit mode, the amount of lock pool space required for six lock requests is 1 KB.
- In the 64-bit mode, the amount of lock pool space required for four lock requests is 1 KB.
- The following formulas can be used to determine the value to be specified for this operand:

HiRDB type	Formula
HiRDB/Parallel Server (32-bit mode)	↑*b* ÷ 6↑ (KB)

HiRDB type	Formula
HiRDB/Parallel Server (64-bit mode)	$\uparrow b \div 4 \uparrow$ (KB)

b:

Total number of transaction lock requests to be executed concurrently by the dictionary server. The number of lock requests depends on the SQL. For details about how to determine the total number of lock requests, see *E. Determining the Number of Locked Resources*.

Tuning the specified value

See the usage rate for the locked resources management table (`% OF USE LOCK TABLE`) displayed for the dictionary server in the statistical information on system operation by the statistics analysis utility. If the maximum usage rate equals or exceeds 80%, it is recommended that the operand's value be increased in preparation for future database expansion. If the maximum usage rate does not exceed 10%, it is recommended that the operand's value be decreased to conserve shared memory space.

Note

If the value specified for this operand is too small, an SQL may return an error.

13) pd_lck_until_disconnect_cnt = *total-number-of-tables-and-RDAREAs-to-be-locked-until-disconnect-specification*

~ <unsigned integer>((0-140000))

Specifies the number of tables and RDAREAs to be held across transactions and the number of resources to be locked for inner replica configuration management, and replica group configuration management. According to the value specified for this operand, tables, RDAREAs, and the blocks for which lock with `UNTIL DISCONNECT` is specified in inner replica configuration management and replica group configuration management are allocated in the shared memory.

Specification guidelines

Normally, this operand need not be specified. Specification of a value other than the default value may be necessary in the following cases:

- When the number of utilities to be executed concurrently increases
- When a holdable cursor is used

For the method of estimating the specification value for this operand, see *D.4 Formula for determining total number of tables and RDAREAs per server locked with UNTIL DISCONNECT specified (pd_lck_until_disconnect_cnt)*.

Tuning the specified value

If the value specified for this operand is small, a transaction may roll back or a utility may terminate abnormally with return code `8`. In such cases, the message `KFPA11914-E` or `KFPH28001-E` is output. If this occurs, the value of this operand should be increased.

When the value of this operand is increased, the amount of required memory space increases proportionately. The required memory size can be expressed as follows: *value-of-this-operand* × 48 (64 in the 64-bit mode) bytes.

14) pd_max_open_holdable_cursors = *maximum-number-of-holdable-cursors-that-can-be-concurrently-open-when-LOCK-statement-with-UNTIL-DISCONNECT-specification-is-not-executed*

~ <unsigned integer>((16-1024))

When you use holdable cursors for a table for which a `LOCK` statement with `UNTIL DISCONNECT` specification is not executed, this operand specifies the maximum number of holdable cursors that can be concurrently open for each transaction.

Note

Specifying a value other than the default value for this operand increases the amount of shared memory used.

Relationship to other operands

The values specified for this operand and the following operands are used for computing the shared memory size for lock servers. For a 32-bit mode HiRDB system, if the values specified for these operands are too large, the shared memory size of the lock servers exceeds 2 GB, and as a result, HiRDB may not start. Therefore, adjust the values specified for these operands so that the shared memory size of the lock servers does not exceed 2 GB.

- `pd_max_access_tables`
- `pd_max_users`
- `pd_max_dic_process`
- `pd_lck_hash_entry`
- `pd_lck_pool_size`

For details about shared memory, see the *HiRDB Version 8 Installation and Design Guide*.

15) pd_lck_hash_entry = *lock-pool-hash-entry-count*

~ <unsigned integer> ((0-2147483647))

Specifies the number of hash table entries to be used in the lock pool. According to the value specified here, HiRDB allocates a lock pool in the shared memory for the unit controller.

Specification guidelines

1. Normally, omit this operand.
2. Consider specifying a value for this operand in the following cases:

- If you do not wish to change the shared memory size if possible when upgrading to Version 06-02 or a newer version, specify `11261`. In this case, the same number of hash entries is allocated as in the older version, the hash table size inside the lock pool remains the same as before.
- Specifying for this operand a value greater than the recommended value may improve performance. However, specifying a value greater than variable *a* described in *Determining the recommended value* does not improve performance over a case in which *a* is specified.

Operand rules

- If `0` is specified for this operand or if this operand and the `pd_lck_hash_entry` operand of each server definition are omitted, HiRDB computes a recommended value for each server. For details about recommended values, see *Determining the recommended value* for the `pd_lck_hash_entry` operand in the server common definition.
- If `0` or a non-prime number is specified for this operand, HiRDB assumes that the largest prime number not exceeding the specified value has been specified.

Note

If the value specified for this operand is too small, hash entry shortage may occur, resulting in performance degradation. If this operand is omitted, neither hash entry shortage nor performance degradation due to hash entry shortage occurs.

Determining the recommended value

For the recommended values, see *Determining the recommended value* for the `pd_lck_hash_entry` operand of the server common definition.

16) pd_dbsync_lck_release_count = *global-buffer-lock-release-interval-during-synchronization-point-processing*

~ <unsigned integer>((0, 100-1073741824))

Specifies an interval for unlocking global buffers, when global buffer locking occurs during synchronization point processing.

During synchronization point processing, search processing occurs on the buffers (update buffers) that must be applied to the disk. Normally, global buffers are unlocked at a specific interval during search processing on the update buffers.

For example, if 100 is specified in this operand, a global buffer is unlocked once when search processing on 100 sectors (global buffer sectors) is completed. After that, the global buffer is locked again and search processing is resumed. In this example, unlocking occurs once every 100 sectors.

Advantage

By specifying this operand, you can adjust the global buffer lock time during synchronization point processing. When a small value is specified in this operand, the global buffer lock time becomes short and transaction performance may improve during synchronization point processing.

To obtain the global buffer pool lock time, execute the statistics analysis utility and in the global buffer pool statistical information check the item called `Buffer pool lock time during synchronization point processing (SYNCL).`

Specification guidelines

Normally, there is no need to specify this operand. Consider specifying this operand when both the following conditions apply:

- Transaction performance drops during synchronization point processing.
- A large number of buffer sectors is specified in the `-n` option of the `pdbuffer` operand.

Operand rules

- If the specified value is in the range 1 to 99, 100 is set automatically.
- If 0 is specified, global buffers are locked until update buffer search processing is completed.

Notes

If a small value is specified in this operand, the update buffer search time becomes longer due to interruptions by other transactions. The global buffers updated during that time are also output during synchronization point processing. Therefore, the number of update buffers to be output during synchronization point processing increases. To obtain the number of update buffers to be output during synchronization point processing, execute the statistics analysis utility and in the global buffer pool statistical information check the item called `Number of synchronization point output` `pages (SYNCW).`

7.3.5 Operands related to buffers

17) pd_sql_object_cache_size = *SQL-object-buffer-size*

~ **<unsigned integer> (KB)**

- 32-bit mode: **((22-256000))**
- 64-bit mode: **((22-2000000))**

Specifies in kilobytes the size of the buffer area (shared memory) in which SQL objects are to be placed.

Specification guidelines

- SQL objects are saved in a buffer until the user's transaction has terminated. The buffer must be large enough to store the SQL objects of all transactions that will be executed concurrently.
- SQL analysis can be reduced by saving the SQL objects of static SQLs in the buffer after transaction termination (until the buffer runs out of space) and sharing them among multiple users who execute the same UAP. To effectively utilize the buffer, it should be allocated so that the SQL objects of frequently-used UAPs are resident in the buffer.
- To estimate the buffer size, first determine the buffer size needed for UAP execution from the length of the SQL objects from the SQL statements to be issued by the UAP. Then, compute the buffer size by considering the number of UAPs that will be executed concurrently and the number of concurrently executing users.
- For details about how to estimate the length of the SQL object from a single SQL statement, see *D.2 Formulas for determining size of SQL object buffer (pd_sql_object_cache_size).*

Tuning the specified value

For details about how to tune the SQL object buffer size, see the *HiRDB Version 8 System Operation Guide*.

7.3.6 Operands related to shared memory

18) pd_dic_shmpool_size = *dictionary-server-shared-memory-size*

~ **<unsigned integer> (KB)**

- 32-bit mode: **((1-200000))**
- 64-bit mode: **((1-4000000000))**

Specifies the size of the area (units: KB) to be used by a dictionary server as part of the shared memory for the unit controller.

Specification guideline

Normally, omit this operand. HiRDB computes a value for this operand if the `pd_dic_shmpool_size` operand of the server common definition and dictionary server definition is omitted. HiRDB computes this value based on *Formula for computing the shared memory used by a single server* in the manual *HiRDB Version 8 Installation and Design Guide*. If you change the value of the operand in the explanation of the variables in this formula, HiRDB automatically re-computes the value for the `pd_dic_shmpool_size` operand. Note that the value of the `pd_assurance_index_no` operand is assumed for the *total-number-of-indexes-inside-server* variable inside the formula. In the formula, for the variables *Number of global buffer pools for index* and *Total number of global buffers (number of pdbuffer operands)*, 500 is assumed in the 32-bit mode and 1000 is assumed in the 64-bit mode.

Tuning the specified value

If any of the following messages is output, increase the specification value of this operand.

- `KFPA20003-E`
- `KFPD00005-E`
- `KFPD00012-E`
- `KFPD00021-E`
- `KFPH20003-E`

Notes

- If an unnecessarily large value is specified for this operand, too large a shared memory area will be allocated. If the value specified for this operand is too small, the following may occur:
  - Unit will not activate
  - A UAP or utility will not execute
- If you omit this operand, make sure that you specify an appropriate value in the `pd_assurance_index_no` operand.

7.3.7 Operands related to the RPC trace information

19) pd_rpc_trace = Y | N

Specifies whether or not RPC trace information is to be collected. HiRDB maintenance information is output in the RPC trace information.

Normally, this operand should be omitted.

Y: Collect RPC trace information.

N: Do not collect RPC trace information.

Note

Specifying Y for this operand degrades communication performance.

20) pd_rpc_trace_name = "*name-of-RPC-trace-collection-file*"

~ <pathname of up to 254 characters>

Specifies an `absolute pathname` for the filename for the RPC trace files. Three RPC trace files are created, with 1, 2, and *l* suffixed to the specified file name.

Note

Files with a maximum size of `pd_rpc_trace_size` *value* × 2 are created under the directory specified by this operand. Attention should be paid to the file capacity.

21) pd_rpc_trace_size = *RPC-trace-collection-file-size*

~ <unsigned integer> ((1024-2147483648)) (Bytes)

Specifies in bytes the size of the RPC trace files.

The value specified by this operand determines the size of both RPC trace file 1 and RPC trace file 2.

Specification guidelines

An RPC trace collects in a single file in time order the information on the communications among all processes. If the volume of this information exceeds the specified value, trace data output is switched to the other file, which means that older trace information will be overwritten. When this happens, the amount of trace information that is available may be inadequate, making troubleshooting difficult. For this reason, at least 1,000,000 should be specified for this operand.

Note

The value specified by this operand does not apply to RPC trace file *l*, because its size is fixed at 0 bytes.

7.3.8 Operands related to troubleshooting information

22) pd_module_trace_max = *maximum-number-of-module-traces-that-can-be-stored*

~ <unsigned integer>((126-16383))

A HiRDB process records the history of the executed functions and macros inside the process private memory. This history is called a *module trace*. This operand specifies the number of module trace records. The content of this history is loaded into the `core` file and is output when a process error occurs.

Specification guidelines

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

Process private memory of the following size is allocated to each process:

In the 32-bit mode: 64 + 48 × *value of* `pd_module_trace_max` *operand* (bytes)

In the 64-bit mode: 64 + 64 × *value of* `pd_module_trace_max` *operand* (bytes)

23) pd_module_trace_timer_level = 0 | 10 | 20

Specifies how to acquire the time to be output in module traces. The following table explains the meaning of the value specified for this operand.

Specified value	Time acquisition method
`0`	Time is output in seconds at every module trace output location.
`10`	Time is output in microseconds only at performance-critical module trace output locations, such as those before and after input/output processing, and time is output in seconds at other locations.
`20`	Time is output in microseconds at every module trace output location.

Specification guidelines

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

If you specify a value other than `0` for this operand, a function for acquiring time in microseconds is issued, and as a result, system performance may decline.

7.3.9 Operands related to global buffers

24) pd_max_add_dbbuff_no =

maximum-global-buffers-count-for-dynamic-addition

~ <unsigned integer>((1-32752))

In order to change global buffers dynamically, this operand specifies the maximum number of global buffers (per server) that can be added dynamically by the `pdbufmod` command.

Condition

`Y` must be specified in the `pd_dbbuff_modify` operand.

Specification guidelines

- Estimate the number of global buffers to be added dynamically by the `pdbufmod` command and then specify a sufficient value based on that value.
- Determine the operand's value in such a manner that the following condition is satisfied:

  Value of `pd_max_add_dbbuff_no` ≤ 2000000 - number of global buffers allocated per server during HiRDB startup

Notes

Do not specify an unnecessarily large value in this operand. If this operand's value is too large, the shared memory used by HiRDB increases, which may result in a shortage of shared memory and an inability of HiRDB to start.

Relationship to other operands

This operand is related to the following operands:

- `SHMMAX`
- `pdbuffer`
- `pd_max_add_dbbuff_shm_no`

25) pd_max_add_dbbuff_shm_no =
maximum-shared-memory-segments-count-for-dynamic-addition

~ <unsigned integer>((1-32752))

In order to change global buffers dynamically, specifies the maximum number of shared memory segments (per server) that can be allocated when dynamic addition is performed by the `pdbufmod` command.

Condition

`Y` must be specified in the `pd_dbbuff_modify` operand.

Specification guidelines

Estimate the number of global buffers to be added dynamically by the `pdbufmod` command and then specify an appropriate value.

Notes

- If the following condition is satisfied, the value of the `pd_max_add_dbbuff_no` operand is assumed in this operand:

  Value of `pd_max_add_dbbuff_shm_no` < value of `pd_max_add_dbbuff_no`

  The value of the `pd_max_add_dbbuff_no` operand is also assumed when the default value satisfies the above condition.
- Do not specify an unnecessarily large value in this operand. If this operand's value is too large, the shared memory used by HiRDB increases, which may result in a shortage of shared memory and an inability of HiRDB to start.
- If the size of a shared memory segment to be added exceeds the `SHMMAX` operand value, shared memory is divided into multiple segments based on the `SHMMAX` operand value as the maximum value. Either increase the `SHMMAX` operand value based on the size of the shared memory segment to be added or increase the `pd_max_add_dbbuff_no` operand value so that no shortage occurs when the shared memory is segmented.
- If you have changed the value of this operand, re-evaluate the maximum size of a shared memory segment in the OS parameters, the maximum number of shared memory segments in the system, and the maximum number of shared memory segments per process.

Relationship to other operands

This operand is related to the following operands:

- `SHMMAX`
- `pdbuffer`
- `pd_max_add_dbbuff_no`

7.3.10 Operands related to Java

Java operands are specified when a Java stored procedure or Java stored function is used. For details about Java stored procedures and Java stored functions, see the *HiRDB Version 8 UAP Development Guide.*

Note that some platforms on which HiRDB is run do not support Java stored procedures and Java stored functions. For details, see *Environment supporting Java stored procedures and functions* in the *HiRDB Version 8 System Operation Guide*.

26) pd_java_stdout_file = *"Java-virtual-machine-standard-output-and-standard-error-output-destination-file"*

~ <pathname>

Specifies as an absolute pathname the file to which the standard output and standard error output are to be output in a Java virtual machine. If this operand is omitted, the standard output and standard error output of the Java virtual machine are ignored.

Specification guideline

Because the size of the file specified by this operand is extremely large, this operand is not normally specified. It is recommended that this operand be specified during debugging of a Java stored procedure or Java stored function. There is no limit to the size of the file that can be specified by this operand.

Note

If there are simultaneous writing attempts from multiple processes, their output contents cannot be guaranteed.

Operand rules

- Up to 255 characters can be used for the pathname.

7.3.11 Operands related to system log files

27) pd_log_dual = Y | N

Specifies whether or not dual system log files are to be used.

`Y`: Use dual system log files.

`N`: Do not use dual system log files.

Advantages

When dual system log files are used, HiRDB collects the same system log information in both files, which are called File A and File B. If a failure occurs in one of the files while the collected system log file is being loaded, the file can be loaded from the other file, resulting in higher system reliability.

Relationship to other operands

When use of dual system log files is specified, the name of the File B system log file must be specified with the `pdlogadpf` operand.

28) pd_log_dual_write_method = serial | parallel

This operand is applicable only to the AIX 5L version of HiRDB.

This operand specifies whether to use the facility for parallel output of system logs. For details about this facility, see the *HiRDB Version 8 Installation and Design Guide*.

`serial`: The facility for parallel output of system logs is not used.

`parallel`: The facility for parallel output of system logs is used.

When `parallel` is specified for this operand, HiRDB uses the Asynchronous I/O Subsystem to execute parallel output to the system log files.

Conditions

When you specify `parallel` for this operand, the following conditions must be satisfied:

- `pd_log_dual = Y`
- The system log files are located in character special files.

If either of these conditions is not satisfied, the facility for parallel output of system logs is not used regardless of the value specified for this operand.

Additionally, when you specify `parallel` for this operand, the Asynchronous I/O Subsystem must be installed.

29) pd_log_remain_space_check = warn | safe

Specifies the processing to be performed by HiRDB when the available space in the system log file falls below the warning level. This is called the facility for monitoring the free area for the system log file. For details on this facility, see the *HiRDB Version 8 System Operation Guide*.

`warn`:

When the available space in the system log file falls below the warning level, the `KFPS01162-W` message is output.

`safe`:

When the available space in the system log file falls below the warning level, scheduling of new transactions is suppressed, all transactions inside the server are forcibly terminated, and the `KFPS01160-E` message is output.

Specification guideline

Hitachi recommends the specification of `safe` because it can reduce the probability of abnormal termination of units due to system log file space shortage. However, when `safe` is specified, all transactions inside the server are forcibly terminated when a shortage occurs in the available space in the system log file. Therefore, the design of the system log file requires more accuracy. For details on system log file design, see the *HiRDB Version 8 Installation and Design Guide*.

30) pd_log_auto_unload_path = "*unload-log-file-output-directory*"[,"*unload-log-file-output-directory*"]...

~ <pathname>((1-136 characters))

Specifies as absolute pathnames the unload file output directories when the automatic log unloading facility is to be used for the system log. A HiRDB file system area name must be specified to create the unload log file in a HiRDB file system area. *The directories or HiRDB file system areas specified for this operand must be created before HiRDB is started.*

Additionally, in this operand, specify a different directory or HiRDB file system area for each server.

For details about the automatic log unloading facility, see the *HiRDB Version 8 System Operation Guide*.

Specification guidelines

It is important to check the available disk space before specifying a directory, so as to ensure that the created unload log file does not cause a disk space shortage.

If an unload log file cannot be created in the specified directory because of a disk space shortage, the automatic log unloading facility stops. If this is a possibility, creation of multiple directories is recommended.

Note, however, that the database recovery operation of selecting the unload files needed for recovery is simplified somewhat when only one directory is used.

Also keep in mind the following when multiple directories are created:

- It is recommended that directories be specified in different partitions to protect against disk errors.
- If the unload log file cannot be created in a single directory because of a full disk or disk error, create an unload log file under a different directory. HiRDB uses the directories specified by this operand in the order of their specification.

Operand rules

- Up to 128 directories can be specified.
- When multiple directories are specified, the same pathname cannot be specified.

Notes

The automatic log unloading facility cannot be used in the following cases:

- N is specified in the pd_log_unload_check operand.
- The log application site uses Real Time SAN Replication based on the log-only synchronous method (the KFPS04689-W message is displayed).

31) pd_log_singleoperation = Y | N

This operand is applicable when dual system log files are used; it need not be specified when dual system log files are not used.

Specifies whether or not the single-operation mode is to be used for the system log files. Even if an error occurs in a system log file, making no dual system log files available, HiRDB (or a unit for a HiRDB/Parallel Server) can continue processing using the remaining single normal system log file without being abnormally terminated. This is called *single operation of the system log files*.

The mode in which continuation of processing is permitted only with both system log files available (normal operation mode) is called *double operation of the system log files*.

Y: Use single operation of the system log files.

N: Do not use single operation of the system log files. Both system log files must always be used.

Condition

This operand is valid only when pd_log_dual = Y is specified.

32) pd_log_rerun_reserved_file_open = Y | N

Specifies whether or not a system log file is to be opened automatically.

If no system log file that can be overwritten is available during a unit restart, HiRDB opens a reserved file (if one is available), makes it overwritable, and continues processing. This is called *automatic opening of system log file*.

A reserved file is used in the following cases:

- Between the time of a restart and the time when the first synchronization point dump is collected
- When none of the opened file groups can be overwritten

Y: Open a system log file automatically (open and use a reserved file).

N: Do not open a system log file automatically (do not use a reserved file).

Advantages

When Y is specified, the unit can be restarted as long as a reserved file is available, even though no file that can be swapped in is available during the unit restart.

However, if a file that is in unload wait status is available, the unit is stopped; once that file has been unloaded, the unit can be restarted.

33) pd_log_rerun_swap = Y | N

Specifies whether or not the system log files are to be swapped during a unit restart.

`Y`: Swap the system log files.

`N`: Do not swap the system log files.

Advantage

When `Y` is specified, there can be a physical separation between the system log files used before and after a restart. Therefore, the system log file that was being used before the restart can be reused during server operation.

34) pd_log_swap_timeout = *wait-time-for-completion-of-system-log-file-swapping*

～ <unsigned integer>((1-32580)) (seconds)

Specifies in seconds the wait time during which swapping of system log files must be completed. If a swap of system log files is not completed within the specified amount of time, the unit terminates abnormally.

Specification guidelines

Normally, there is no need to specify this operand. If it takes a long time to swap system log files for a reason such as poor machine performance, you can specify this operand with a value that is greater than the default value. To detect errors or delays quickly during system log file swapping so that the unit can be terminated (such as because of a disk failure), reduce the value of this operand.

35) pd_log_unload_check = Y | N

Specifies whether or not HiRDB is to check the unload status of system log files.

`Y`:

Check the unload status (normal operation).

`N`:

Do not check the unload status. A system log file is placed in swappable status, regardless of its unload status, when both of the following conditions are satisfied:

- It is in overwritable status

- It is in extraction completed status (HiRDB Datareplicator)
- Overwriting permitted status for online reorganization (HiRDB Staticizer Option)

In such a case, the system log file operation method is to release checking of unload status. For details about this operation method, see the *HiRDB Version 8 System Operation Guide*.

Advantages

Specifying `N` provides the following advantages:

- Operations are simplified because the system log file unload operation is eliminated.
- It is not necessary to provide files for storing unload files.

Specification guideline

`N` should be specified if the system log file will not be needed for database recovery (i.e., if recovery from a backup collection point will be sufficient).

Notes

The following points apply when `N` is specified:

- Database can be recovered only if backups have been made.
- If this option is specified when the system log file is required for database recovery, it will not be possible to recover the database.

36) pd_log_max_data_size = *log-input/output-buffer-size*

~ <unsigned integer>((32000-523000)) (Bytes)

Specifies in bytes the size of the buffer to be used for system log input/output operations.

Specification guideline

Specify a value that satisfies conditional expression 1 below. If `uap` is specified in the `pd_rpl_reflect_mode` operand or a recovery-unnecessary front-end server is used, specify a value that satisfies both the conditional expressions below (1 and 2). To optimize the value, use the tuning method for the specification value.

Conditional expression 1: log input/output buffer length ≥ *a*

a: 72 x total number of back-end servers + 1432

Conditional expression 2: log input/output buffer length ≥ *b*

b: (Maximum number of back-end servers subject to update processing by a single transaction + 1) x 128 + 64

Tuning the specified value

A value (other than the default value) may need to be specified for this operand after the following types of statistics analysis utility statistical information related to system operation have been checked:

- Number of buffer sectors waiting for input/output (`# OF BUFFER FOR WAIT I/O`)

  If the average number of buffer sectors waiting for input/output significantly exceeds 100, increase the value for this operand so that the average approaches 100.

- Number of waits caused by lack of a current buffer (`# OF WAIT THREAD`)

  If the number of waits caused by lack of a current buffer is not 0, increase the value for this operand.

Relationship to other operands

Use this operand and the `pd_log_write_buff_count` operand to determine the log output buffer size.

37) pd_log_write_buff_count = *log-output-buffer-sectors-count*

～ <unsigned integer>((3-65000))

Specifies the number of buffer sectors to be used for system log output.

Tuning the specified value

A value (other than the default value) may need to be specified for this operand after the following types of statistics analysis utility statistical information related to system operation have been checked:

- Number of times the buffer was full (`# OF BUFFER FULL`)
- Number of waits caused by lack of a current buffer (`# OF WAIT THREAD`)

If either of these values is large, a larger value should be specified in order to improve throughput.

Relationship to other operands

Use this operand and the `pd_log_max_data_size` operand to determine the log output buffer size.

38) pd_log_rec_leng = *system-log-file-record-length*

～ <unsigned integer>((1024, 2048, 4096)) (Bytes)

Specifies the record length for the system log files; the specifiable values are

1024, 2048, and 4096.

The record length specified in the `-l` option of the `pdloginit` command should be specified for this operand.

Notes

- If a value that is different from the record length specified by the `-l` option of the `pdloginit` command is specified for this operand, system log files cannot be opened.
- For details about how to modify the system log file record length, see the *HiRDB Version 8 System Operation Guide*.

7.3.12 Operands related to synchronization point dump files

39) pd_spd_dual = Y | N

Specifies whether to use dual synchronization point dump files.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for the server are also used at the log application site.

`Y`: Uses dual synchronization point dump files.

`N`: Does not use dual synchronization point dump files.

Advantage

When dual synchronization point dump files are used, HiRDB collects the same synchronization point dump in both dump files A and B. Even when an error occurs in one of the files when the collected synchronization point dump is being loaded, the synchronization point dump can be loaded from the other file, thus improving system reliability.

Relationship to other operands

To use dual synchronization point dump files, specify the name of synchronization point dump file B in the `pdlogadpf` operand.

40) pd_spd_assurance_msg = Y | N

Specifies whether or not the `KFPS02183-I` message is to be output when a synchronization point dump is completed.

`Y`: Output the message.

`N`: Do not output the message.

41) pd_spd_assurance_count = *number-of-guaranteed-valid-generations*

～ <unsigned integer>((1-2))

Specifies the range of system log files to be saved during HiRDB operation as a protection against events such as a synchronization point dump file input error during system recovery. What is specified here is a number of synchronization point dump file generations; the specified value is referred to as the *number of guaranteed-valid generations*. The number of past generations of synchronization point dump files specified here are overwrite disabled.

If Real Time SAN Replication based on the log-only synchronous method is used, this operand is ignored, if specified, at the log application site. The system log files used at the log application site are overwritten by the transaction execution site even if they include system log information that has not been applied.

Advantage

When `2` is specified as the number of guaranteed-valid generations and an error occurs in the most recent synchronization point dump file generation, the system can be recovered using the preceding synchronization point dump file generation, resulting in higher reliability.

Specification guideline

- To improve reliability, `2` should be specified for the number of guaranteed valid generations. However, the number of overwrite disabled synchronization point dump files increases (to two).
- If reliability has been improved with the use of dual synchronization point dump files, Hitachi recommends that you omit this operand or specify `1` for it.

Notes

- The minimum number of synchronization point dump files needed is *number-of-guaranteed-valid-generations* + 1.
- Specifying `2` increases the number of overwrite disabled synchronization point dump files. Moreover, the system log files that correspond to the overwrite disabled synchronization point dump files are also overwrite disabled. Consequently, specifying `2` for the number of guaranteed valid generations increases the number of overwrite disabled system log files. As a result, a shortage may occur in the number of system log files that can be swapped in. To prevent this, it may be necessary to reevaluate the system log file capacity.

42) pd_spd_reduced_mode = *reduced-mode-operation-option*

~ <unsigned integer>((0-2))

Specifies whether or not the reduced mode operation for synchronization point dump files is to be used.

Reduced mode operation is a facility for continuing processing if at least two synchronization point dump files can be used, even if an event such as a file error during HiRDB operation or restart reduces the number of synchronization point dump files to or below the number of required files (*number of guaranteed valid generations*[*] + 1).

[*] Value specified for the `pd_spd_assurance_count` operand.

`0`: Do not use the reduced mode operation.

`1`: Use the reduced mode operation.

`2`: Use the reduced mode operation and issue a warning message whenever a synchronization point dump is being acquired during the reduced mode operation.

43) pd_spd_reserved_file_auto_open = Y | N

Specifies whether or not a synchronization point dump file is to be opened automatically. When `Y` is specified and an error in a synchronization point dump file reduces the number of synchronization point dump files to the number of files necessary for operation (*number of guaranteed valid generations*[*] + 1), HiRDB opens a reserved file (if one is available), makes it overwritable, and continues processing. This is called *automatic opening of synchronization point dump file*.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for the server are also used at the log application site.

[*] Value specified for the `pd_spd_assurance_count` operand.

`Y`:

Open a synchronization point dump file automatically. A reserved file is opened when the number of files falls below the number necessary for operation (*number of guaranteed valid generations* + 1).

`N`:

Do not open a synchronization point dump file. No reserved file is opened, even though the number of files falls below the number necessary for operation (*number of guaranteed valid generations* + 1).

Relationship to other operands

This operand has a higher priority than the `pd_spd_reduced_mode` operand.

44) pd_spd_max_data_size = *synchronization-point-dump-file-buffer-size*

~ <unsigned integer>((32000-4000000)) (Bytes)

Specifies in bytes the size of the buffer (shared memory) to be used for synchronization point dump file input/output operations.

The value specified here affects the number of synchronization point dump file input/output operations.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for server are also used at the log application site.

Specification guidelines

- Normally, this operand need not be specified.
- When the value of this operand is increased, the number of synchronization point dump file input/output operations is reduced.

45) pd_log_sdinterval = ***system-log-output-volume*****[,interval]**

Specifies the collection interval for synchronization point dumps. The following should be taken into consideration in specifying this operand:

- Volume of system log information output since the previous synchronization point
- Amount of time that has elapsed since the previous synchronization point

system-log-output-volume**: ~ <unsigned integer>((100-100000)) (Number of log blocks)**

Specifies an interval between synchronization point dumps in terms of the number of blocks of log information. A synchronization point dump is collected each time system log information equivalent to the number of log blocks specified here has been output.

interval**: ~ <unsigned integer>((0 or 10-1440)) (Minutes)**

Specifies a synchronization point dump collection interval in terms of number of minutes between synchronization point dumps.

- When `0` is specified for the interval, HiRDB does not use a time interval for collecting synchronization point dumps.
- If no transactions execute during an interval, no synchronization point dump is collected even though the amount of the time specified here has elapsed.

Specification guidelines

- This operand need not be specified if no specific amount of time is specified for restarting HiRDB.
- The value specified for this operand affects the amount of time required to restart HiRDB.
- Specifying a small value for this operand reduces the amount of time

required for database recovery during a HiRDB restart. However, because the frequency of synchronization point dumps increases, online performance may deteriorate in some cases.

Conversely, specifying a large value for this operand increases the amount of time required for database recovery during a HiRDB restart. However, because the frequency of synchronization point dumps decreases, online performance may improve in some cases.

Tuning the specified value

The synchronization point dump collection interval can be checked with the statistics analysis utility; the relevant information is shown under `SYNC POINT GET INTERVAL` in the statistical information related to system operation. The average `SYNC POINT GET INTERVAL` value should be used. If the synchronization point dump collection interval is determined to be too long, the specification value should be decreased; conversely, if it is determined to be too short, the specification value should be increased.

Note

The synchronization point dump collection interval is determined on the basis of the volume of system log information that is output. Therefore, committing data from memory to a database takes a long time during intervals that have few updating transactions. If an error occurs at such a time, it will take longer to recover the transactions that were generated during that period. If this is a possibility, the synchronization point dump collection interval should be set by also using the `interval` value.

7.3.13 Operands related to server status files

46) pd_sts_file_name_1 = "*logical-file-name*", "*file-a-status-file-name*", "*file-b-status-file-name*"

: ...

pd_sts_file_name_7 = "*logical-file-name*", "*file-a-status-file-name*", "*file-b-status-file-name*"

Define server status files. Although the `pd_sts_file_name_2` to `7` operands may be omitted, the `pd_sts_file_name_1` operand cannot be omitted.

"*logical-file-name*" ~ <identifier>((1-8 characters))

Specifies the logical file name of a status file for the dictionary server.

When a command that manipulates the status file is to be executed, the logical file name defined here must be specified.

"*file-a-status-file-name*" ~ <pathname>((up to 167 characters))

Specifies the name of the File A status file as an absolute pathname.

"*file-b-status-file-name*" ~ **<pathname>((up to 167 characters))**

Specifies the name of the File B status file as an absolute pathname.

Specification guidelines

- The files specified as File A and File B should be status files created with the `pdstsinit` command.
- If a file that has not been created with the `pdstsinit` command is specified, a virtual status file is created.
- If an error occurs in a status file, HiRDB swaps the status files. If no spare file is available for swapping, HiRDB (or a unit for a HiRDB/ Parallel Server) terminates abnormally. For this reason, defining a large number of system files improves system reliability, but at the expense of increasing the amount of disk space that is required.
- The status files specified for File A and File B must have the same record length and capacity.

Operand rules

- Up to seven instances of this operand may be specified.
- A status file has a dual structure consisting of File A and File B; both must be specified.
- Environment variables cannot be used for the absolute pathnames of the File A and File B file names.
- The same names cannot be specified for the logical file name, the File A file name, and the File B file name.

Notes

- When HiRDB is started normally, the primary file (the file that was the primary file when HiRDB was terminated) is inherited. However, if there is no current file that can be inherited, for example because all status files have been initialized, the first status file that was specified among those specified by the `pd_sts_file_name_1` to `7` operands becomes the current file, and the remaining files become spare files if they can be opened. Any files that cannot be opened become reserved files.
- When HiRDB is restarted, the primary file (the file that was the primary file when HiRDB was terminated) is inherited.

Using a virtual status file

Specifying a virtual status file makes it possible to add a new status file while

HiRDB is running.

For example, if the number of spare files is reduced because of errors in status files, virtual status files can be converted into spare files. The procedure for converting virtual status files into spare files follows.

Procedure

1. Use the `pdstsinit` command to create a status file in a HiRDB file system area for system files.
2. Use the `pdstsopen` command to open the status file.

These operations can be performed while HiRDB is running; there is no need to stop HiRDB.

- **Advantages and disadvantages**

  Defining a virtual status file reduces the amount of space required in the HiRDB file system area. However, a virtual status file cannot be added as a spare file if the HiRDB file system area for system files does not have sufficient space (sufficient space for adding the file), thus reducing system reliability.

  When virtual status files are not defined, the amount of space required in the HiRDB file system area is increased. However, because a swap destination is guaranteed when a file error occurs, system reliability is increased.

- **Notes**

  When virtual status files are defined, HiRDB determines that a status file error has occurred when HiRDB is started. For this reason, HiRDB cannot start if `stop` (default value) is specified for the `pd_sts_initial_error` operand. When virtual status files are defined, specify `continue` or `excontinue` for the `pd_sts_initial_error` operand. It is also necessary before starting HiRDB to specify the current file in the `pd_sts_last_active_file` operand.

Relationship to other operands

This operand is related to the `pd_sts_subfile_name_1` to `7` operands.

47) pd_sts_subfile_name_1 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"

:

pd_sts_subfile_name_7 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*

","*secondary-status-file-name-for-log-application-processing*"

Defines the server status files for log application processing, which are used at the log application site when Real Time SAN Replication based on the log-only synchronous method is used.

"*logical-file-name*": ~ <identifier> ((1-8 characters)

Specifies the logical file name of a server status file for log application processing. This must be a logical file name specified in `pd_sts_file_name_1` through `pd_sts_file_name_7`. A logical file name defined here is used for executing commands that manipulate the status files for log application processing.

"*primary-status-file-name-for-log-application-processing*": ~ <path name> ((up to 167 characters)

Specifies the absolute path name of a primary status file for log application processing.

"*secondary-status-file-name-for-log-application-processing*": ~ <path name> ((up to 167 characters)

Specifies the absolute path name of a secondary status file for log application processing.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.

Specification guidelines

- For the primary and secondary status file names for log application processing, specify the names of status files for log application processing that were created in the preparations for log application. If any other status file for log application processing is specified, the specified file becomes a *virtual status file for log application processing*.
- If an error occurs in a status file for log application processing, HiRDB swaps status files for log application processing. If no spare file is available for swapping, HiRDB (or the unit in the case of a HiRDB/Parallel Server) terminates abnormally. For this reason, defining a large number of status files improves system reliability, but at the expense of increasing the amount of disk space that is required.

- The specified primary and secondary status files for log application processing must have the same record length and capacity.

Operand rules

- You can specify a maximum of 7 instances of this operand.
- This operand is ignored at the transaction execution site.
- If you performed the preparations for log application during HiRDB startup, create status files for log application processing that correspond to all status files for transaction processing that can be opened and that were specified in `pd_sts_file_name_1` through `pd_sts_file_name_7`.
- The system uses dual status files for log application processing. Make sure that you specify both files.
- Environment variables cannot be used for the absolute path names of the primary and secondary status files for log application processing.
- Each logical file name, primary status file name for log application processing, and secondary status file name for log application processing must be unique.
- The names of the primary and secondary status files for log application processing must be different from the primary and secondary status file names specified in the `pd_sts_file_name_1` through `pd_sts_file_name_7` operands.

Notes

When HiRDB is started normally or restarted, the current file (the file that was the current file when HiRDB last terminated) is inherited. However, if there is no current file that can be inherited, such as when all status files for log application processing have been initialized, the log application site can no longer be started. If this occurs, perform the preparations for log application.

How to use a virtual status file for log application processing

See the description of the `pd_sts_file_name_1` through `pd_sts_file_name_7` operands.

Relationship to other operands

This operand is related to the `pd_sts_file_name_1` through `pd_sts_file_name_7` operands.

7.3.14 Operands related to server status files (when an error occurs)

For details about the measures to be taken when an error occurs in a status file, see the

HiRDB Version 8 System Operation Guide.

48) pd_sts_initial_error = stop | continue | excontinue

When a server (single server, front-end server, dictionary server, or back-end server) starts, HiRDB performs a process of identifying the current server status file. This operand specifies the action that HiRDB takes when any of the following errors are detected during this identification process.

- No real server status file is found.
- An error is detected in the server status file.

The current file identification process is applied to the server status files specified by the `pd_sts_file_name_1` to `7` operands.

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing specified in the `pd_sts_subfile_name1` through `pd_sts_subfile_name7` operands are also used at the log application site. For the log application site, replace the `pd_sts_last_active_file` and `pd_sts_last_active_side` operands in the above description with the `pd_sts_last_active_subfile` and `pd_sts_last_active_side_sub` operands, respectively.

`stop`:

When an error is detected in a server status file during the current file identification process, the startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started. In this case, first take a corrective action for the status file in which the error was detected, and then start HiRDB.

`continue` or `excontinue`:

Even when an error is detected in the server status file during the current file identification process, the startup of the server is continued if the current file is normal. However, the startup may be stopped depending on the value specified for the `pd_sts_singleoperation` operand (whether operation should continue with a single status file). The following table shows the relationship to the `pd_sts_singleoperation` operand.

- **Relationship to the pd_sts_singleoperation operand**

pd_sts_singleoperation operand value	Processing by HiRDB	HiRDB administrator's action
`continue`	When an error is detected in a server status file, HiRDB cannot identify the current file, and thus startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.	The HiRDB administrator identifies the current file and specifies the `pd_sts_last_active_file` and `pd_sts_last_active_side` operands. Afterwards, start HiRDB.
`stop` (default value)	When an error is detected in a server status file, HiRDB identifies the current file and startup of the server is continued. However, if the primary and secondary files satisfy any of the conditions listed in the table below (cases in which HiRDB cannot identify the current file), startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.	If HiRDB cannot identify the current file, the HiRDB administrator identifies the current file and specifies the `pd_sts_last_active_file` and `pd_sts_last_active_side` operands. Afterwards, start HiRDB.

- **When HiRDB cannot identify the current file**

pd_sts_initial_error operand value	File A status	File B status
`continue`	Error shutdown	Error shutdown
	Error shutdown	Open (initial state)
	Error shutdown	No real file
	Open (initial state)	Error shutdown
	Open (initial state)	No real file
	No real file	Open (initial state)
	No real file	Error shutdown
	No real file	No real file

pd_sts_initial_error operand value	File A status	File B status
excontinue	Error shutdown	Error shutdown
	Error shutdown	No real file
	No real file	Error shutdown
	No real file	No real file

Therefore, to minimize actions by the HiRDB administrator (to reduce the number of cases in which both the pd_sts_last_active_file and pd_sts_last_active_side operands must be specified), specify the following:

- pd_sts_initial_error = excontinue
- pd_sts_singleoperation = stop

Specification guidelines

The following table shows the specification guidelines, along with their advantages and disadvantages.

Item	pd_sts_initial_error operand value	
	stop	continue or excontinue
Processing by HiRDB during server startup	When an error is detected in a server status file, startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.	Even when an error is detected in a server status file, the startup of the server is continued if the current file is normal.
Specification guideline	To improve system reliability, specify stop.	To simplify the error-handling actions during HiRDB startup, specify continue or excontinue.
Advantage	Guarantees that all server status files of the server are normal when the server starts. Therefore, if an error occurs in the current file after HiRDB has started, it can be swapped to a spare file.	Even when an error is detected in a server status file during server startup, HiRDB can start with the remaining normal files only. Therefore, HiRDB stop time can be shortened. In this case, because the number of spare files has become small, it is necessary to immediately repair the status files containing errors.
Disadvantage	Possibility increases that an error in a server status file stops the startup of HiRDB.	Because HiRDB may be running with only a small number of spare files, system reliability is low. Depending on the number of spare files available, it may not be possible to swap server status files.

Notes

- If both current files are abnormal, startup of the server is stopped regardless of the value specified for this operand, and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.
- Before starting HiRDB, do not initialize the current file with the `pdstsinit` command. If the current file is initialized, HiRDB cannot be restarted.

Remarks

For details about the operand value, HiRDB processing, and HiRDB administrator's action, see *Remarks* in the section on the `pd_sts_initial_error` operand.

49) pd_sts_singleoperation = stop | continue

Specifies whether or not processing may continue in the single-operation mode for server status files.

The status file single-operation mode means that processing is to continue using only the normal file (single file) when an error occurs in the status file and a spare file is not available. For details about the status file single-operation mode, see the *HiRDB Version 8 Installation and Design Guide*.

If an error occurs in one of the current files and a spare file is available, the status file is swapped and processing continues regardless of the value specified for this operand (operation in the single-operation mode does not occur).

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing are also used at the log application site.

`stop`:

Do not permit operation in the single-operation mode. If operation in the single-operation mode is necessary, HiRDB (or a unit for a HiRDB/Parallel Server) is abnormally terminated. If HiRDB is abnormally terminated, allocate spare files and then restart HiRDB.

`continue`:

Enable operation in the single-operation mode. When the single-operation mode goes into effect, the message `KFPS01044-I` is output. If an error occurs in the normal file during operation in the single-operation mode or if HiRDB (or a unit for a HiRDB/Parallel Server) is abnormally terminated while the status file is being updated, HiRDB cannot be restarted. Therefore, when the single-operation mode goes into effect, immediately allocate spare files.

Specification guidelines

- Hitachi recommends that you specify `stop` to increase system reliability. Hitachi also recommends that you increase the number of spare files to guard against errors in the current files.
- The following table shows the specification guidelines, along with their advantages and disadvantages.

Item	pd_sts_singleoperation operand value	
	stop	continue
Specification guideline	To improve system reliability, specify `stop`.	Specify `continue` if it is important not to stop HiRDB.
Advantage	When an error occurs in one of the current files and a spare file is not available, operation in the single-operation mode does not occur and HiRDB is abnormally terminated. Consequently, the possibility of losing the content of the current files is reduced.	Even when an error occurs in one of the current files and a spare file is not available, processing can be continued. Therefore, the possibility that an error in the status file stops HiRDB is reduced.
Disadvantage	Possibility that an error in the status file stops HiRDB increases. However, increasing the number of spare files can reduce this possibility.	If an error occurs in the normal status file during operation in the single-operation mode or if HiRDB is abnormally terminated during updating of the status file, the content of the current file is lost and HiRDB cannot be restarted.

Relationship to other operands

The combination of the values specified for the `pd_sts_singleoperation` and `pd_sts_initial_error` operands determines the processing by HiRDB when an error occurs in the status file. Therefore, the values to be specified for these operands should be determined together.

50) pd_sts_last_active_file = "*logical-file-name*"

~ <identifier> ((1-8 characters))

Specifies the name of the logical file to be used as the current status file at the time of the front-end server startup. HiRDB compares the file name specified in this operand with the file name selected by HiRDB to be the current file. If the file names match, HiRDB (or a unit for a HiRDB/Parallel Server) is started; otherwise, HiRDB (or a unit for a HiRDB/Parallel Server) is not started.

Conditions

The following conditions must be satisfied:

- continue or excontinue is specified in the pd_sts_initial_error operand.
- It cannot be determined if the current file selected by the HiRDB system was the most recent current file during the previous session.

Specification guidelines

1. To start HiRDB immediately after initializing all status files:

   From among the operable logical files specified in the pd_sts_file_name1 to 7 operands, specify the one that has the smallest number. Forced startup will be used in this case, regardless of the previous termination mode.

2. When both of the current files are normal:

   Specify the name of the current file.* If HiRDB cannot be started even though the name of the current file is specified, the current file may have been initialized. In this case, first initialize all status files, then use the method in 1 above to start HiRDB (forced startup will be used, regardless of the previous termination mode).

3. When one of the current status files has an error:

   Use the method in 2 previously, with the following operands specified:

   - pd_syssts_singleoperation = continue
   - pd_sts_last_active_side

4. When both of the current status files have errors:

   Initialize all status files, then execute the method in 1 above (forced startup will be used, regardless of the previous termination mode).

5. When a virtual status file is specified

   Specify the name of the current file.*

* The names of the current files (that were active at the end of the previous operation) can be determined from the following messages:

- KFPS01001-I
- KFPS01010-E
- KFPS01011-I
- KFPS01063-I

Of the status files displayed by these messages, the one that is reported in the message that was output most recently is the current file.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for transaction processing are used at the transaction execution site. At the log application site, this operand is ignored, if specified.

51) pd_sts_last_active_side = A | B

Specify this operand if you want to start the dictionary server when one of the current files is in an error state. Specify the normal status file for this operand. HiRDB compares the file specified in this operand with the file selected by HiRDB. If they match, HiRDB copies the contents of the normal status file to secondary File A and File B, then HiRDB switches the spare file in as the current file and starts the unit. If the files do not match, HiRDB (or a unit for a HiRDB/ Parallel Server) is not started.

Conditions

The following operands must be specified:

- `pd_sts_initial_error = continue` or `excontinue`
- `pd_sts_last_active_file`

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for transaction processing are used at the transaction execution site. At the log application site, this operand is ignored, if specified.

52) pd_sts_last_active_subfile = "*logical-file-name*"

~ <identifier> ((1-8 characters))

When Real Time SAN Replication based on the log-only synchronous method is used, this operand specifies the logical file name of the status file for log application processing that is to be used as the current file when HiRDB (or the unit in the case of a HiRDB/Parallel Server) starts. HiRDB compares the file specified in this operand with the current file selected by HiRDB. If the file names match, HiRDB starts; if not, HiRDB does not start. Note that this operand is ignored if HiRDB is started immediately after the database is inherited pursuant to execution of the `pdrisedbto` command.

Conditions

For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*

The following conditions must be satisfied:

- Y must be specified in the pd_rise_use operand, and also syssync must be specified in the pd_rise_pairvolume_combination operand.
- continue or excontinue is specified in the pd_sts_initial_error operand.
- It cannot be determined if the current file selected by the HiRDB system was the most recent current file during the previous session.

Specification guidelines

1. When starting HiRDB immediately after initializing all status files for log application processing

   Perform the preparations for log application. Once the preparations for log application have been completed successfully, start HiRDB without specifying this operand.

2. When both of the current files are normal

   Specify the name of the current file.# If HiRDB cannot be started even though the name of the current file is specified, the current file may have been initialized. In such a case, first initialize all status files for log application processing, then use the method in 1 above to start HiRDB. Note that in this case, forced startup will be used, regardless of the previous termination mode.

3. When one of the current files has an error

   Use the method in 2 above and specify the following operands:

   - Specify continue in the pd_syssts_singleoperation operand
   - Specify the pd_sts_last_active_side_sub operand

4. When both of the current files have errors

   Perform the preparations for log application. Once the preparations for log application have been completed successfully, start HiRDB without specifying this operand.

5. When a virtual status file for log application processing is specified

   Specify the name of the current file.#

#

The names of the current files (that were active at the end of the previous session) can be determined from the following messages:

- KFPS01001-I

- KFPS01010-E
- KFPS01011-I
- KFPS01063-I

Of the status files for log application processing that are displayed in these messages, the one that is reported in the message that was output most recently is the current file.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the status files for log application processing for the server are used at the log application site. At the transaction execution site, this operand is ignored, if specified.

53) pd_sts_last_active_side_sub = A | B

When Real Time SAN Replication based on the log-only synchronous method is used, specify this operand if you wish to start HiRDB (or the unit in the case of a HiRDB/Parallel Server) when one of the current files is in error status. In such a case, you specify the normal file in this operand. HiRDB compares the file specified in this operand with the file selected by HiRDB. If their names match, HiRDB copies the contents of the normal file into the secondary file. HiRDB then designates the secondary file as the current file and starts itself. If the file names do not match, HiRDB does not start. This operand is ignored if HiRDB is started immediately after the database was inherited pursuant to execution of the `pdrisedbto` command.

Conditions

For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*

The following conditions must be satisfied:

- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.
- `continue` or `excontinue` is specified in the `pd_sts_initial_error` operand.
- The `pd_sts_last_active_subfile` operand is specified.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing are used at the log application site. At the transaction execution site, this operand is ignored, if specified.

7.3.15 Operands related to work table files

54) pdwork -v "*HiRDB-file-system-area-name*"[,"*HiRDB-file-system-area-name*"]...

~ <pathname of up to 141 characters>

This operand must be specified in order to execute SQL statements.

Specifies the names of HiRDB file system areas for work table files. Work table files are used for temporary storage of information during execution of SQL statements; they are created automatically by HiRDB. For details about the SQL statements that require a work table file, see the *HiRDB Version 8 Description*.

Notes

- Specify in this operand the HiRDB file system area that was initialized using the `pdfmkfs` command.
- If the size of the work table file is large, specify a large HiRDB file system area. For the method of estimating the work table file size, see the *HiRDB Version 8 Installation and Design Guide*.
- The HiRDB file system areas for work table files must be different from the HiRDB file system areas for system files and RDAREAs.
- In the Linux version, you cannot specify a HiRDB file system area created in a character special file.

Operand rules

- At least one HiRDB file system area must be specified.
- A maximum of 16 HiRDB file system areas may be specified.
- This operand may be specified only once in the dictionary server definition. If it is specified more than once, the first specification is effective.
- When you are specifying multiple HiRDB file system areas, those created in character special files can coexist with those created in regular files.

7.3.16 Operands related to system log file configuration

55) pdlogadfg -d sys -g *file-group-name* [ONL]

Specifies a file group for a system log file. This operand cannot be omitted and must be specified. Use the `pdlogadpf` operand to assign system log files to the file group specified here.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name* ~ <identifier>((1-8 characters))

Specifies a name for the file group. All file group names must be unique within a server.

`ONL`:

Specify this option to enable (open) this file group when HiRDB is running. You can specify 2 to 200 file groups with `ONL` specified.

Operand rule

This operand must be specified at least twice but no more than 200 times.

Notes

Before adding, modifying, or deleting this operand when a HiRDB Datareplicator (extracted side) is linked, terminate the corresponding HiRDB Datareplicator. If this operand is modified while the HiRDB Datareplicator is running, its extraction process may fail.

56) pdlogadpf -d sys -g *file-group-name* -a "*system-log-file-name*" [-b "*system-log-file-name*"]

Specifies the system log files that comprise the file group. This operand cannot be omitted; it must be specified once for each file group.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name* ~ <identifier>((1-8 characters))

Specifies the file group name specified by the `pdlogadfg` operand. The file group name must be unique within the unit.

-a "*system-log-file-name*" ~ <pathname>((up to 167 characters))

Specifies an absolute path name as the name of the system log file that comprises the file group. Specify the name of the system log file that was initialized using the `pdloginit` command. Note that the system log file name must be unique within the unit.

-b "*system-log-file-name*" ~ <pathname>((up to 167 characters))

Specifies an absolute path name as the name of the system log file B when dual system log files are to be used (`pd_log_dual = Y` specified). If `pd_log_dual = Y` is not specified, the system log file name is invalid, even if it is specified.

Specify the name of the system log file that was initialized using the `pdloginit` command. Note that the system log file name must be unique within the unit.

Notes

Before adding, modifying, or deleting this operand when HiRDB Datareplicator (extracted side) is linked, terminate the corresponding HiRDB Datareplicator. If this operand is modified while HiRDB Datareplicator is running, its extraction process may fail.

7.3.17 Operands related to synchronization point dump file configuration

57) pdlogadfg -d spd -g *file-group-name* [ONL]

Specifies a file group for synchronization point dump files. This operand cannot be omitted and must be specified. Use the `pdlogadpf` operand to assign synchronization point dump files to the file group specified here.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name* ~ <<identifier>((1-8 characters))

Specifies a name for the file group. All file group names within a unit must be server.

`ONL`:

Specify this option to enable (open) this file group when HiRDB is running. You can specify 2 to 30 file groups with `ONL` specified.

Operand rule

This operand must be specified at least twice but no more than 60 times.

58) pdlogadpf -d spd -g *file-group-name* -a "*synchronization-point-dump-file-name*" [-b "*synchronization-point-dump-file-name*"]

Specifies the synchronization point dump files that comprise the file group. This operand cannot be omitted; it must be specified once for each file group.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name*: ~ <identifier>((1-8 characters))

Specifies the file group name specified by the `pdlogadfg` operand. The file group name must be unique within the unit.

-a "*synchronization-point-dump-file-name*": ~ <pathname> ((up to 167 characters))

Specifies an absolute path name as the name of the synchronization point dump file that comprises the file group. Specify the name of the synchronization point dump file that was initialized using the `pdloginit` command. Note that the synchronization point dump file name must be unique within the unit.

-b "*synchronization-point-dump-file-name*": ~ <pathname> ((up to 167 characters))

Specifies an absolute path name as the name of the synchronization point dump file B when dual synchronization point dump files are to be used (`pd_spd_dual = Y` specified). If `pd_spd_dual = Y` is not specified, the synchronization point dump file name is invalid, even if it is specified.

Specify the name of the synchronization point dump file that was initialized using the `pdloginit` command. Note that the synchronization point dump file name must be unique within the unit.

59) pdlogadfg -d ssp -g *file-group-name*[ONL]

When Real Time SAN Replication based on the log-only synchronous method is used, this operand specifies a group of synchronization point dump files for log application processing that is to be used at the log application site. The `pdlogadpf` operand is used to allocate the synchronization point dump files for log application processing.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name*: ~ <identifier> ((1-8 characters))

Specifies the name of a file group. This file group name must correspond to a set of synchronization point dump files for transaction processing.

`ONL`:

Specifies that this file group is to be made available (opened) when HiRDB is running. You can specify 2 to 30 file groups with `ONL` specified.

If Real Time SAN Replication based on the log-only synchronous method is used, when HiRDB is started for the first time at the log application site after the preparations for log application have been performed, the file groups that need to be inherited from the transaction execution site become available when the single server starts, regardless of the specification of `ONL`.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.

Operand rules

You can specify 2 to 60 instances of this operand.

Notes

Specify a `pdlogadfg -d ssp` operand for each file group specified in a `pdlogadfg -d spd` operand.

60) pdlogadpf -d ssp -g *file-group-name* -a "*synchronization-point-dump-file-name-for-log-application-processing*" [-b "*synchronization-point-dump-file-name-for-log-application-processing*"]

Specifies synchronization point dump files for log application processing that constitute a file group. You must specify one `pdlogadpf -d ssp` operand for each file group.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name*: ～ <identifier> ((1-8 characters))

Specifies the name of a file group specified in a `pdlogadfg` operand. This file group name must correspond to a set of primary synchronization point dump files for transaction processing.

-a "*synchronization-point-dump-file-name-for-log-application-processing*": ～ <path name> ((up to 167 characters))

Specifies the absolute path name of a synchronization point dump file for log application processing that is to be included in the file group. This must be the name of a synchronization point dump file for log application processing that was created during the preparations for log application. Note that each synchronization point dump file name for log application processing must be unique within the same unit.

-b "*synchronization-point-dump-file-name-for-log-application-processing*": ～ <path name> ((up to 167 characters))

When dual synchronization point dump files for log application processing are used (`Y` is specified in the `pd_spd_dual` operand), this option specifies the absolute path name of a secondary synchronization point dump file for log application processing. If `Y` is not specified in the `pd_spd_dual` operand, the `-b` option is ignored, if specified. You must specify in this option the name of a synchronization point dump file for log application processing created during the preparations for log application. Note that each synchronization point dump file for log application processing must be unique within the same unit.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.

Notes

Specify a `pdlogadpf -d ssp` operand for each file group specified in a `pdlogadfg -d ssp` operand.

7.3.18 Operands related to Plug-ins

61) pdplgprm -n *plug-in-name* [-s *shared-memory-size*]

Specifies the name of a plug-in and the size of the memory to be shared by the plug-in. This operand should be omitted if no plug-ins are to be used or no plug-ins will run in dictionary server.

Conditions

The plug-in specified here must have been registered in HiRDB with the `pdplgrgst` command.

-n *plug-in-name*: ~ <identifier>((1-30 characters))

For details about the names of plug-ins that can be specified here, see the manuals for the plug-ins.

-s *shared-memory-size*: ~ <unsigned integer> ((1-2000000)) <<0>> (KB)

Specifies in kilobytes the size of the shared memory to be used by the plug-in. For details about the size of the shared memory to be used by the plug-in, see the manual for the applicable plug-in.

Chapter

8. Back-End Server Definition

This chapter explains the operands of the back-end server definition.

This chapter contains the following sections:

8.1 Operand formats

A back-end server definition defines information for a back-end server. This section explains the formats used to specify the operands of a back-end server definition. Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *8.2 Operands whose specification values can be changed* and *8.3 Operand explanations*.

For users who are creating HiRDB system definitions for the first time

The first step is to determine the values to be specified for the operands shown in boldface type. In principle, HiRDB can be started once the boldface operands have been specified.

No.	Format	Operand category
1	`[set pd_max_bes_process =` *maximum-number-of-activated-processes-per-back-end-server*`]`*	Processes
2	`[set pd_process_count =` *resident-processes-count* `[,`*resident-processes-count-at-server-startup*`]]`*	
3	`[set pd_server_cleanup_interval =` *interval-for-stopping-nonresident-server-processes*`]`*	
4	`[set pd_svr_castoff_size =` *maximum-memory-size-used-by-server-process*`]`*	
5	`[set pd_max_open_fds =` *maximum-number-of-files-and-pipes-accessed-by-process*`]`	
6	`[set pd_max_ard_process =` *asynchronous-READ-process-count*`]`*	
7	`[set pd_dfw_awt_process =` *number-of-processes-to-be-written-in-parallel-in-deferred-write*`]`	
8	`[set pd_work_buff_mode = each | pool]`*	Work tables
9	`[set pd_work_buff_size =` *work-table-buffer-size*`]`*	
10	`[set pd_work_buff_expand_limit =` *work-table-buffer-expansion-limit*`]`*	
11	`[set pd_spd_syncpoint_skip_limit =` *maximum-number-of-skipped-synchronization-point-dumps*`]`*	System monitoring

No.	Format	Operand category
12	[set pd_dfw_syncpoint_skip_limit = *maximum-number-of-skipped-synchronization-point-dumps-resulting-from-delay-of-synchronization-point-dump-acquisition-due-to-deferred-write-processing*]*	
13	[set pd_lck_pool_size = *server-lock-pool-size*]*	Lock
14	[set pd_lck_until_disconnect_cnt = *total-number-of-tables-and-RDAREAs-to-be-locked-per-server-UNTIL-DISCONNECT-specification*]*	
15	[set pd_max_open_holdable_cursors = *maximum-number-of-holdable-cursors-that-can-be-concurrently-open-when-LOCK-statement-with-UNTIL-DISCONNECT-specification-is-not-executed*]	
16	[set pd_lck_hash_entry = *lock-pool-hash-entry-count*]*	
17	[set pd_dbsync_lck_release_count = *global-buffer-lock-release-interval-during-synchronization-point-processing*]*	
18	[set pd_sql_object_cache_size = *SQL-object-buffer-size*]*	Buffers
19	[set pd_bes_shmpool_size = *back-end-server-shared-memory-size*]*	Shared memory
20	[set pd_rpc_trace = Y | N]*	RPC trace information
21	[set pd_rpc_trace_name = "*name-for-RPC-trace-collection-files*"]*	
22	[set pd_rpc_trace_size = *RPC-trace-collection-file-size*]*	
23	[set pd_module_trace_max = *maximum-number-of-module-traces-that-can-be-stored*]*	Troubleshooting information
24	[set pd_module_trace_timer_level = *module-trace-output-time-acquisition-method*]	
25	[set pd_max_add_dbbuff_no = *maximum-global-buffers-count-for-dynamic-addition*]*	Global buffer
26	[set pd_max_add_dbbuff_shm_no = *maximum-shared-memory-segments-count-for-dynamic-addition*]*	

No.	Format	Operand category
27	`[set pd_plugin_ixmk_dir =` "*index-information-file-creation-directory-name*" *or* "*index-information-file-creation-HiRDB-file-system-area-name*"]*	Delayed batch creation of plug-in index
28	`[set pd_java_stdout_file =` "*Java-virtual-machine-standard-output-or-standard-error-output-destination-file*"]*	Java
29	`[set pd_foreign_server_libpath =` *foreign-server-client-library-path-name* [,"*foreign-server-client-library-path-name*"]	HiRDB External Data Access facility
30	`[set pd_log_dual = Y | N]`*	System log files
31	`[set pd_log_dual_write_method = serial | parallel]`	
32	`[set pd_log_remain_space_check = warn | safe]`*	
33	`[set pd_log_auto_unload_path =` "*unload-log-file-output-directory*" *[,*"*unload-log-file-output-directory*"]...]	
34	`[set pd_log_singleoperation = Y | N]`*	
35	`[set pd_log_rerun_reserved_file_open = Y | N]`*	
36	`[set pd_log_rerun_swap = Y | N]`*	
37	`[set pd_log_swap_timeout =` *wait-time-for-completion-of-system-log-file-swapping*]*	
38	`[set pd_log_unload_check = Y | N]`*	
39	`[set pd_log_max_data_size =` *log-input/output-buffer-size*]*	
40	`[set pd_log_write_buff_count =` *log-output-buffer-sectors-count*]*	
41	`[set pd_log_rec_leng =` *system-log-file–record-length*]*	

No.	Format	Operand category
42	`[set pd_spd_dual = Y	N]`*	Synchronization point dump files
43	`[set pd_spd_assurance_msg = Y	N]`*	
44	`[set pd_spd_assurance_count =` *number-of-guaranteed-valid-generations*`]`*	
45	`[set pd_spd_reduced_mode =` *reduced-mode-operation-option*`]`*	
46	`[set pd_spd_reserved_file_auto_open = Y	N]`*	
47	`[set pd_spd_max_data_size =` *synchronization-point-dump-file-buffer-size*`]`*	
48	`[set pd_log_sdinterval =` *system-log-output-volume*`[,`*interval*`]]`*	
49	`set pd_sts_file_name_1 =` "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"	Server status files
	`[set pd_sts_file_name_2 =` "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"]	
	`[set pd_sts_file_name_3 =` "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"`]`	
	`[set pd_sts_file_name_4 =` "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"`]`	
	`[set pd_sts_file_name_5 =` "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"`]`	
	`[set pd_sts_file_name_6 =` "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"`]`	
	`[set pd_sts_file_name_7 =` "*logical-file-name*" ,"*file-a-status-file-name*","*file-b-status-file-name*"`]`	
50	`[set pd_sts_subfile_name_1 =` "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"`]`	
	`[set pd_sts_subfile_name_2 =` "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"`]`	

No.	Format	Operand category
	[set pd_sts_subfile_name_3 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_4 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_5 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_6 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
	[set pd_sts_subfile_name_7 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"]	
51	[set pd_sts_initial_error = stop | continue | excontinue]*	Server status files (when an error occurs)
52	[set pd_sts_singleoperation = stop | continue]*	
53	[set pd_sts_last_active_file = "*logical-file-name*"]	
54	[set pd_sts_last_active_side = A | B]	
55	[set pd_sts_last_active_subfile = "*logical-file-name*"]	
56	[set pd_sts_last_active_side_sub = A | B]	
57	[set pd_bes_connection_hold = Y | N]	BES connection holding facility
58	[set pd_bes_conn_hold_trn_interval = *back-end-server-connection-hold-time*]	
59	pdwork -v "*HiRDB-file-system-area-name*"[,"*HiRDB-file-system-area-name*"]...	Work table files

No.	Format	Operand category
60	{{pdlogadfg -d sys -g *file-group-name* [ONL]}}	System log file configuration
61	{{pdlogadpf -d sys -g *file-group-name* -a "*system-log-file-name*" [-b "*system-log-file-name*"]}}	
62	{{pdlogadfg -d spd -g *file-group-name* [ONL]}}	Synchronization point dump file configuration
63	{{pdlogadpf -d spd -g *file-group-name* -a "*synchronization-point-dump-file-name*" [-b "*synchronization-point-dump-file-name*"]}}	
64	{{pdlogadfg -d ssp -g *file-group-name* [ONL]}}	
65	{{set pdlogadpf -d ssp -g *file-group-name* -a "*synchronization-point-dump-file-name-for-log-application-processing*" [-b "*synchronization-point-dump-file-name-for-log-application-processing*"]}}	
66	{{[pdplgprm -n *plug-in-name* [-s *shared-memory-size*]]}}	Plug-ins
67	[putenv *environment-variables-needed-by-foreign-server*]	HiRDB External Data Access facility (environment variables)

* If this operand is omitted, the value specified in the same operand in the server common definition is used.

8.2 Operands whose specification values can be changed

The values of some of the back-end server definition operands can be changed in the system common definition, unit control information definition, and server common definition. These operands are indicated below. After a planned termination, forced termination, or abnormal termination of HiRDB, some HiRDB system definition operands can be modified while others cannot be modified. The operands that can be modified are indicated below.

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
1	pd_max_bes_process	—	—	Y	—	—	Y	N	Y2
2	pd_process_count	—	—	Y	Y	Y	Y	Y2	Y2
3	pd_server_cleanup_interval	—	—	Y	Y	Y	Y	Y	Y
4	pd_svr_castoff_size	—	—	Y	Y	Y	Y	Y	Y
5	pd_max_open_fds	—	—	—	—	—	Y	Y	Y
6	pd_max_ard_process	—	—	Y	—	Y	Y	Y	Y
7	pd_dfw_awt_process	—	—	Y	—	Y	Y	Y	Y
8	pd_work_buff_mode	—	—	Y	—	Y	Y	Y	Y
9	pd_work_buff_size	—	—	Y	—	Y	Y	Y	Y
10	pd_work_buff_expand_limit	—	—	Y	—	Y	Y	Y	Y
11	pd_spd_syncpoint_skip_limit	—	—	Y	Y	Y	Y	Y	Y

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
12	`pd_dfw_syncpoint_skip_limit`	—	—	Y	—	Y	Y	Y	Y
13	`pd_lck_pool_size`	—	—	Y	—	Y	Y	Y[2]	Y
14	`pd_lck_until_disconnect_cnt`	—	—	Y	—	Y	Y	N	Y
15	`pd_max_open_holdable_cursors`	—	—	Y	—	Y	Y	N	N
16	`pd_lck_hash_entry`	—	—	Y	Y	Y	Y	Y	Y
17	`pd_dbsync_lck_release_count`	—	—	Y	—	Y	Y	Y	Y
18	`pd_sql_object_cache_size`	Y	—	Y	Y	Y	Y	N	Y
19	`pd_bes_shmpool_size`	—	—	Y	—	—	Y	Y[2]	Y[2]
20	`pd_rpc_trace`	Y	Y	Y	Y	Y	Y	Y	Y
21	`pd_rpc_trace_name`	Y	Y	Y	Y	Y	Y	Y	Y
22	`pd_rpc_trace_size`	Y	Y	Y	Y	Y	Y	Y	Y
23	`pd_module_trace_max`	Y	Y	Y	Y	Y	Y	Y	Y
24	`pd_module_trace_timer_level`	Y	Y	Y	Y	Y	Y	Y	Y

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
25	`pd_max_add_dbbuff_no`	—	—	Y	—	Y	Y	N	Y
26	`pd_max_add_dbbuff_shm_no`	—	—	Y	—	Y	Y	N	Y
27	`pd_plugin_ixmk_dir`	—	—	—	—	—	Y	N	Y
28	`pd_java_stdout_file`	Y	Y	Y	Y	Y	Y	Y	Y
29	`pd_foreign_server_libpath`	—	—	—	—	—	Y	Y	Y
30	`pd_log_dual`	—	—	Y	Y	Y	Y	N	N
31	`pd_log_dual_write_method`	—	—	Y	Y	Y	Y	Y	Y
32	`pd_log_remain_space_check`	—	—	Y	Y	Y	Y	Y	Y
33	`pd_log_auto_unload_path`	—	—	—	Y	Y	Y	N	N
34	`pd_log_singleoperation`	—	—	Y	Y	Y	Y	N	N
35	`pd_log_rerun_reserved_file_open`	—	—	Y	Y	Y	Y	Y	Y
36	`pd_log_rerun_swap`	—	—	Y	Y	Y	Y	Y	Y
37	`pd_log_swap_timeout`	—	—	Y	Y	Y	Y	Y	Y
38	`pd_log_unload_check`	—	—	Y	Y	Y	Y	Y	Y

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
39	pd_log_max_data_size	--	--	Y	Y	Y	Y	Y[2,3]	Y[2]
40	pd_log_write_buff_count	--	--	Y	Y	Y	Y	Y[3]	Y
41	pd_log_rec_leng	--	--	Y	Y	Y	Y	Y	Y
42	pd_spd_dual	--	--	Y	Y	Y	Y	N	N
43	pd_spd_assurance_msg	--	--	Y	Y	Y	Y	Y	Y
44	pd_spd_assurance_count	--	--	Y	Y	Y	Y	N	N
45	pd_spd_reduced_mode	--	--	Y	Y	Y	Y	N	Y
46	pd_spd_reserved_file_auto_open	--	--	Y	Y	Y	Y	N	Y
47	pd_spd_max_data_size	--	--	Y	Y	Y	Y	N	N
48	pd_log_sdinterval	--	--	Y	Y	Y	Y	Y	Y
49	pd_sts_file_name_1-7	--	--	--	Y	Y	Y	Y[1]	Y[1]
50	pd_sts_subfile_name_1-7	--	--	--	Y	Y	Y	N	N
51	pd_sts_initial_error	--	--	Y	Y	Y	Y	Y	Y
52	pd_sts_singleoperation	--	--	Y	Y	Y	Y	Y	Y

No.	Operand	Definition name						Modifiable after forced or abnormal termination?	Modifiable after planned termination?
		SYS	UNT	SVR	FES	DS	BES		
53	`pd_sts_last_active_file`	—	—	—	Y	Y	Y	Y	Y
54	`pd_sts_last_active_side`	—	—	—	Y	Y	Y	Y	Y
55	`pd_sts_last_active_subfile`	—	—	—	Y	Y	Y	Y	Y
56	`pd_sts_last_active_side_sub`	—	—	—	Y	Y	Y	Y	Y
57	`pd_bes_connection_hold`	—	—	Y	—	—	Y	Y	Y
58	`pd_bes_conn_hold_trn_interval`	—	—	Y	—	—	Y	Y	Y
59	`pdwork`	—	—	—	—	Y	Y	N	N
60	`pdlogadfg -d sys`	—	—	—	Y	Y	Y	Y[1]	Y[1]
61	`pdlogadpf -d sys`	—	—	—	Y	Y	Y	Y[1]	Y[1]
62	`pdlogadfg -d spd`	—	—	—	Y	Y	Y	Y[1]	Y[1]
63	`pdlogadpf -d spd`	—	—	—	Y	Y	Y	Y[1]	Y[1]
64	`pdlogadfg -d ssp`	—	—	—	Y	Y	Y	Y[1]	Y[1]
65	`pdlogadpf -d ssp`	—	—	—	Y	Y	Y	Y[1]	Y[1]
66	`pdplgprm`	—	—	—	Y	Y	Y	Y	Y

Y: Yes, specification value can be modified.

N: No, specification value cannot be modified.

—: Specification value cannot be modified because operand is not applicable.

SYS: System common definition

UNT: Unit control information definition

SVR: Server common definition

FES: Front-end server definition

DS: Dictionary server definition

BES: Back-end server definition

[1] Operand that can be added only; it cannot be deleted or modified.

[2] If the specified value is too small, it may not be possible to restart HiRDB. When this is the case, restore the value to what it was before the change and then restart HiRDB.

[3] If the specified value is too large, it may not be possible to restart HiRDB. When this is the case, restore the value to what it was before the change and then restart HiRDB.

8.3 Operand explanations

8.3.1 Operands related to processes

1) pd_max_bes_process = ***maximum-number-of-activated-processes-per-back-end-server***

~ <unsigned integer>((1-2048))

Specifies the maximum number of processes that can be activated per back-end server. For multi front-end servers, processes that exceed the value specified in the `pd_max_users` can sometimes become concentrated in a single back-end server. The `pd_max_bes_process` operand specifies the maximum number of processes that can be activated per back-end server when that number exceeds the value of the `pd_max_users` operand.

Condition

This operand is applicable when multiple front-end servers are used.

Specification guidelines

- The value determined by the following formula indicates the maximum number of processes that may become concentrated in a single back-end server.

  `pd_max_users` value × *number of front-end servers*

  Specify a value for the `pd_max_bes_process` operand by using the value determined here as the upper limit and taking the degree of process concentration in a single back-end server into consideration. Specifying an unnecessarily large value may cause memory shortage.
- If the value specified is smaller than the `pd_max_users` value, the `pd_max_users` value is assumed as the default, and a warning message (`KFPS01888-W`) is output.
- If updatable online reorganization is used, specify an operand value that satisfies the following condition. Otherwise, HiRDB cannot be started.

  `pd_max_bes_process` + `pd_max_reflect_process_count` value ≤ 2048

Tuning the specified value

For details about how to tune the maximum number of processes that can be activated, see the *HiRDB Version 8 System Operation Guide*.

2) pd_process_count = ***resident-processes-count*****[,*****resident-processes-count-at-server-startup*****]**

~ <unsigned integer>((0-2048))

resident-processes-count

Specifies the number of processes that can be made resident in the back-end server. A resident process is a process that is activated at the time the server is started.

Advantage

By activating the processes used by transactions that can be processed concurrently by the back-end server at the time of system startup and keeping them resident, the process startup time can be reduced even when new transactions are entered. However, HiRDB startup will take longer.

Specification guidelines

See *Specification guidelines* for the `pd_process_count` operand of the server common definition.

Tuning the specified value

For details about how to tune the resident processes count, see the *HiRDB Version 8 System Operation Guide*.

Notes

- Because the resident processes count has an effect on memory space availability, specifying an unnecessarily large number may prevent HiRDB from starting.
- If more processes than the resident process count are needed, additional processes are dynamically started, up to the maximum processes count allowed. However, depending on the value specified for the `pd_max_server_process` operand, it may not be possible to start all of the processes indicated by the maximum processes count.

Operand rule

See *Operand rules* for the `pd_process_count` operand of the server common definition.

resident-processes-count-at-server-startup

Specifies the number of processes that can be made resident during HiRDB startup.

It is common for resident processes to be activated during HiRDB startup. If there are many resident processes, the amount of time required for HiRDB startup increases proportionately. For example, it takes approximately 1 second to activate a process on a server machine with a 100-MIPS performance rating.

Following are the differences in processing that result depending on whether or not a resident processes count at server startup is specified:

- When there is no specification of a resident processes count at server startup (when, for example, `pd_process_count = 500` is specified)

  All 500 resident processes are activated during HiRDB startup, and HiRDB will not start until they are all activated. In this example, it would take a 100-MIPS server machine about 500 seconds to activate all the resident processes during HiRDB startup.

- When a resident processes count at server startup is specified (when, for example, `pd_process_count = 500, 50` is specified)

  Some of the resident processes (50 in this case) are activated during HiRDB startup, and the remaining resident processes are activated after HiRDB startup. HiRDB can be started as long as the specified number of resident processes is activated. In this example, it would take a 100-MIPS server machine about 50 seconds to activate 50 resident processes during HiRDB startup. The remaining 450 resident processes (in this example) would be activated after HiRDB startup.

Advantage

The amount of time required for HiRDB startup is reduced. You should use this option when you want to reduce the HiRDB startup time as much as possible, such as when you are using the system switchover facility.

Specification guideline

The specified value should be equal to the number of processes that will be required immediately after HiRDB startup is completed.

Notes

When you specify a resident processes count at server startup, you should recheck the value in the `PDCWAITTIME` operand of the client environment definition.

If more UAPs than the resident processes count at server startup are to be executed immediately following HiRDB startup, transaction processing may not be performed until after the remaining resident processes have been activated. Therefore, if the value specified in the `PDCWAITTIME` operand of the client environment definition is small, it may not be possible to process some UAPs due to timeouts. For details about the `PDCWAITTIME` operand, see the *HiRDB Version 8 UAP Development Guide*.

3) pd_server_cleanup_interval = ***interval-for-stopping-nonresident-server-processes***

~ <unsigned integer> ((0-1440)) (minutes)

Specifies in minutes the interval for checking for nonresident server processes in HiRDB that are to be stopped. The facility for stopping nonresident server processes is applied when the number of executing server processes exceeds the number of processes that can be made resident (value specified by the `pd_process_count` operand). The number of server processes that the facility stops is computed automatically by HiRDB.

Advantage

This facility improves the utilization rates of the memory and of process resources because it increases the number of nonresident processes that can be reused when the workload (number of server processes that are executing) peaks.

Specification guidelines

- For example, if there is only one hour a day during which peak workload occurs and the intervals between peaks within that hour are approximately two minutes apart, `2` should be specified for this operand.
- This facility does not have any effect if the number of server processes that execute concurrently during peak periods is always fewer than the number of resident processes. In this case, this operand should be omitted.

Tuning the specified value

Collect statistical information on system operations for each server for one week. Determine the workload peaks from the number of server processes being serviced (`# OF PROCESSES ON SERVICE`). If that peak value exceeds the currently set number of processes that can be made resident (value specified by the `pd_process_count` operand), determine the interval between individual peaks and set that number of minutes.

However, if there are ample resources, such as memory and CPU, in the server machine, adding the shortfall in the number of processes to the number of resident processes (i.e., increasing the value of the `pd_process_count` operand) would be more effective in improving performance than specifying the `pd_server_cleanup_interval` operand.

Note

When this operand is omitted or `0` is specified, the system checks every 10 seconds for nonresident server processes that are waiting to be serviced and stops any such processes that are found.

4) pd_svr_castoff_size = *maximum-memory-size-used-by-server-process*

~ <unsigned integer>((0-2048)) (MB)

This operand need not be specified for the Linux version.

Specifies the maximum size of memory used by each server process processed by a back-end server. If at the applicable trigger point shown below the amount of memory being used by a server process exceeds the value specified here, that server process is terminated. When a server process is so terminated, the `KFPS00350-w` message is output. This is called the *facility for monitoring the memory size of server processes*; for details, see the *HiRDB Version 8 System Operation Guide*.

Server type	Process name	Process termination trigger
Back-end server	pdbes	At transaction decision

Advantages

The facility for monitoring the memory size of server processes resolves the following problem:

- *The memory size of a server-resident process becomes too large during a particular SQL processing, significantly reducing the amount of system memory that is available.*

The HiRDB server releases memory space that is no longer needed. However, even when a program releases memory, the OS holds the memory area itself in the memory management facility inside the applicable process. Consequently, a process that becomes large in terms of using a large memory area even once never shrinks and continues to have an adverse impact on the system, especially for resident processes. The facility for monitoring the memory size of server processes can prevent memory shortages because it terminates even resident processes.

Application criterion

Apply this facility when the amount of memory space used by a HiRDB server process becomes large, resulting in memory shortages.

Specification guidelines

- Maximum value for this operand

  Normally, specify a value by considering the maximum processing capability of HiRDB. Assuming that the maximum number of SQL statements may be concurrently connected, determine for this operand a value that satisfies the following condition in each unit:

  $a \times (b + c) < d$

  a: Number of server processes within the unit

(*maximum-number-of-concurrent-connections* × *server-count-inside-unit*)

b: Virtual memory size for one server process immediately following HiRDB startup

c: `pd_svr_castoff_size` operand value

d: Memory size that can be allocated to a unit (memory size excluding the area used by other programs)

- Minimum value for this operand

  If the value specified for this operand is smaller than the memory size needed for normal SQL processing, the efficiency of making processes resident will deteriorate, resulting in frequent process terminations and restarts. Each time this occurs, a message is output to the syslogfile or the message log file, resulting in further performance degradation. To prevent this, select for this operand a value that satisfies the following condition for each server:

  $a - b < c$

  a: Server process virtual memory size following SQL process termination

  b: Virtual memory size for server process immediately following HiRDB startup

  c: `pd_svr_castoff_size` operand value

The virtual memory size can be determined by an OS command (e.g., the `top` command in HP-UX).

Operand rule

When `0` is specified for this operand, the facility for monitoring the memory size of server processes is not applied.

5) pd_max_open_fds = *maximum-number-of-files-and-pipes-accessed-by-process*

～ <unsigned integer>((1-8192))<<320>>

Specifies the maximum number of files and pipes accessed by HiRDB processes.

Specification guidelines

- Omit this operand if the maximum number of the plug-in index storage RDAREAs updated in a single transaction is less than 100.
- If the maximum number of the plug-in index storage RDAREAs updated in a single transaction is equal to or greater than 100, specify the maximum number + 320 for this operand.

Notes

The maximum value of this operand differs depending on the OS type as follows:

- HP-UX: 8192
- Solaris: 2048
- 64-bit mode Solaris: 8192
- AIX: 8192
- Linux: 1024

If an index information file is created during utility execution, files and pipes are accessed up to the smaller of the following two values regardless of the value specified for this operand:

- The physical upper limit for the number of files that can be opened or locked by a single process of the OS's operating system parameter
- The maximum value of this operand for each OS

The physical upper limit for the number of files that can be opened or locked by a single process of the OS's operating system parameter is described below.

- For HP-UX: Value of `maxfiles_lim`
- For Solaris: Value of `rlim_fd_max`
- For AIX: Value of `nofiles_hard`
- For Linux: Value of `NR_OPEN`

6) pd_max_ard_process = *asynchronous-READ-process-count*

~ <unsigned integer>((0-256))

Specify this operand if you use the asynchronous READ facility. For this operand, specify the number of processes necessary for asynchronous READ operations. For a HiRDB/Parallel Server, the value specified here indicates the number of processes per server (back-end server or dictionary server). For details on the asynchronous READ facility, see the *HiRDB Version 8 Installation and Design Guide*.

Condition

A value of `1` or greater must be specified for the `-m` option of the `pdbuffer` operand.

Advantage

The asynchronous READ facility is especially effective (improves performance) when character special files, for which input and output take a long time, are used. Conversely, when regular files, for which input and output do not take a long time, or SANRISE disks are used, the asynchronous READ facility may not have much effect because of the following reasons:

- Input/out time does not overlap with CPU time most of the time.
- Communication processing has a large overhead.

Specification guidelines

- Specify `0` or `1`. However, if a value between 2 and 256 is specified for the `-m` option of the `pdbuffer` operand, specify the same value as the `-m` option value. If a value greater than 256 is specified for the `-m` option, specify the same value as the number of disk devices that store RDAREAs and system files (the number of such disk devices per server for a HiRDB/Parallel Server) or `256`.
- Increasing the value of this operand can shorten the processing time when the degree of concurrency is high for the SQL statements to which the asynchronous READ facility is applied. Decreasing the value of this operand may lengthen the processing time when the degree of concurrency is high for the SQL statements to which the asynchronous READ facility is applied. This is because asynchronous READ processes may have to wait for processing completion.
- Because a number of processes equaling *value-of-this-operand* × *server-count* are started, determine a value for this operand by taking resources (shared memory and message queue) into consideration. For the method of estimating shared memory and message queue sizes, see the *HiRDB Version 8 Installation and Design Guide*.

Tuning the specified value

For the method of tuning the specification value (the number of asynchronous READ processes), see the *HiRDB Version 8 System Operation Guide*.

Relationship to other operands

If you change the value of this operand, reevaluate the value of the `pd_max_server_process` operand.

Operand rule

If you specify `0` for this operand, the asynchronous READ facility is not used.

7) pd_dfw_awt_process = *number-of-processes-to-be-written-in-parallel-in-deferred-write*

~ <unsigned integer>((2-255))

Specify this operand when you use the *facility for parallel writes in deferred write processing* for all buffer pools. Specify for this operand the number of processes to be processed in parallel. Increasing the number of processes can shorten the write processing time. For details about the *facility for parallel writes in deferred write processing,* see the *HiRDB Version 8 Installation and Design Guide*.

Specification guidelines

Specify `2`, which is the smallest value that enables the facility for parallel writes in deferred write processing. Furthermore, to determine the value for this operand, see *Tuning deferred write processing* in the *HiRDB Version 8 Installation and Design Guide*.

Note

Specifying the facility for parallel writes in deferred write processing increases the number of processes, and consequently raises the CPU usage rate.

8.3.2 Operands related to work tables

8) pd_work_buff_mode = each | pool

Specifies the method of allocating buffers when HiRDB creates tables.

`each`: Allocate a buffer for each work table.

`pool`: Allocate a buffer pool for each server process.

Specification guidelines

- Normally, `pool` is specified. `pool` is the appropriate specification when a large volume of data is to be retrieved and when manipulations such as `join`, `ORDER BY`, and `GROUP BY` are to be performed.
- When the size of the process private area that can be used for work table buffers is predetermined, `pool` should be specified. When `pool` is specified, HiRDB efficiently allocates work table buffers to work tables.

  In such a case, the process private area is occupied on the basis of the value specified in `pd_work_buff_size`, and input/output operations on work tables are buffered in that pool. Therefore, the process private memory is occupied only to the extent of the value specified in `pd_work_buff_size`.

9) pd_work_buff_size = *work-table-buffer-size*

~ <unsigned integer> (KB)

- 32-bit mode: **((128-1000000))**
- 64-bit mode: **((128-4000000000))**

Specifies in kilobytes the size of buffers for work tables to be created by HiRDB.

Item	pd_work_buff_mode=each is specified	pd_work_buff_mode=pool is specified
Advantage	A large work table buffer size reduces the number of I/O operations associated with data manipulation, which means that the execution time of SQLs that use work tables is also reduced. However, because each server's process private memory is used, specification of this option should take into account the overall size of the system memory (real memory and virtual memory). If `pd_work_buff_mode` = each is specified, the memory size to be allocated is *value of* `pd_work_buff_size` × *required number of work tables*. A process private memory that has been allocated will remain allocated to that process at least in the virtual memory (will not be released until that process disappears) even after release is requested by HiRDB. Therefore, specifying an unnecessarily large value may cause a virtual memory shortage for other processes.	
Application criterion	`pd_work_buff_mode = pool` should be specified when a large volume of data is to be retrieved and when manipulations such as `join`, `ORDER BY`, and `GROUP BY` are to be performed.	
Specification guidelines	• Specify the size of the buffer to be allocated for one work table. • If a value greater than the work table memory capacity is specified for the work table buffer size, input/output to the work table file during work table creation is eliminated, thus reducing the time necessary for work tables. The following formula can be used to determine the work table memory capacity: `Work table memory capacity = Applicable work table size* ÷ 2`	• Specify the size of the buffer pool to be allocated for the entire server process. • Specify a value between 4352 and 5120 when a large volume of data is to be retrieved or when manipulations such as join, `ORDER BY`, and `GROUP BY` are to be performed. Specifying such a value increases the unit of sorting input/output, thus reducing the sort time. • If a large value is specified for the work table buffer size, with the total work table memory capacity for each SQL statement as the upper limit, input/output operations on the work table file during work table creation is eliminated, thus reducing the time necessary for work tables. The following formula can be used to determine the total work table memory capacity for each SQL statement: `Total work table memory capacity per SQL statement =` $a \times b + c \times d$

Item	pd_work_buff_mode=each is specified	pd_work_buff_mode=pool is specified
Notes	When multiple users execute processes concurrently or when an SQL statement that uses multiple work tables is executed, the specified size buffer is allocated for each work table. Consequently, specifying a large value may result in a memory shortage.	If the value specified for the work table buffer size is smaller than the number of work tables to be used by each SQL statement, the processing time may become longer than when `each` is specified. Specifically, specify a value that is at least equal to *maximum number of work tables for each SQL statement* × *128*. The following formula can be used to determine the maximum number of work tables for each SQL statement: `Maximum number of work tables for each SQL statement =` *b* `+` *d*
Operand rule	Specify a multiple of 128. If the value is not a multiple of 128, it is rounded up to the next multiple of 128.	• Specify a multiple of 128. If the value is not a multiple of 128, it is rounded up to the next multiple of 128. • Specify at least 384. If a value that is smaller than 384 is specified, it is rounded up to 384.

a:

↑{Capacity of work table (for storing column information)* (KB) ÷ 2} ÷ 128↑ × 128

b:

Maximum number of work tables (for storing column information)*

c:

↑{Capacity of work table (for storing positional information)* (KB) ÷ 2} ÷ 128↑ × 128

d:

Maximum number of work tables (for storing positional information)*

* For details about how to determine these values, see the *HiRDB Version 8 Installation and Design Guide*.

10) pd_work_buff_expand_limit = *work-table-buffer-expansion-limit*

~ <unsigned integer> (KB)

- 32-bit mode: **((128-1000000))**
- 64-bit mode: **((128-4000000000))**

The size of the work table buffer to be created by HiRDB is specified by the pd_work_buff_size operand. Specify the pd_work_buff_expand_limit operand if you want to automatically expand a work table buffer when the space in this buffer becomes insufficient. The work table buffer is expanded up to the size specified by this operand.

For example, when the following values are specified for the operands, a 1,024-KB work table buffer is normally allocated. When this size becomes insufficient, the work table buffer is expanded up to 2,048 KB.

- pd_work_buff_size = 1024
- pd_work_buff_expand_limit = 2048

HiRDB expands a work table buffer in the following cases:

- The necessary work table buffer cannot be allocated when hash execution is applied to an execution method that uses hash join or subquery hash as the joining method.
- A 128-KB work table buffer allocated to each work table becomes insufficient when multiple work tables are concurrently used.

Condition

The pd_work_buff_mode operand must be omitted or pool must be specified for it.

Advantage

You can prevent a work table buffer shortage (too small a value specified for the pd_work_buff_size operand) from causing UAP errors.

Notes

- A work table buffer is not expanded when either of the following conditions is satisfied:
  - The pd_work_buff_expand_limit operand is not specified.
  - pd_work_buff_expand_limit operand value ≤ pd_work_buff_size operand value
- When a work table buffer is expanded, resident processes are also terminated at the following triggers (except in the Linux version) to free up memory:

  If a holdable cursor is not being used, a dictionary server process is terminated when a transaction is normally terminated or cancelled.

  If a holdable cursor is being used, a dictionary server process is terminated when a UAP is disconnected from HiRDB.

Operand rule

Specify a multiple of 128. If a value other than a multiple of 128 is specified, it is automatically rounded up to a multiple of 128.

Relationship to other operands

When a work table buffer is expanded for the first time in a back-end server process, the `KFPH29008-I` message is output. Note that you can use the `pd_work_table_option` operand to suppress this message output.

Note

After a work table buffer has been expanded, when the number of work tables being used by the applicable server process goes to zero, the expanded work table buffer is released. The number of work tables being used can go to zero in the following cases:

- All cursors that were being used are closed. (In this case, the number of work tables being used may not go to zero.)
- A transaction is normally terminated or cancelled when a holdable cursor is not being used.
- A UAP is disconnected from HiRDB when a holdable cursor is being used.

Remarks

Hash join, subquery hash execution is applied in the following cases:

- *Application of optimizing mode 2 based on cost* and *hash join, subquery hash execution* are specified in the `pd_additional_optimize_level` operand, the `PDADDITIONALOPTLVL` operand of the client environment definition, or the `ADD OPTIMIZE LEVEL` operand of the SQL compile option.
- `HASH` is specified for the SQL optimization specification of the joining method inside an SQL statement.
- `HASH` is specified for the SQL optimization specification of the subquery execution method inside an SQL statement.

8.3.3 Operands related to system monitoring

11) pd_spd_syncpoint_skip_limit = *maximum-number-of-skipped-synchronization-point-dumps*

～ <unsigned integer>((0, 2-100000))

If an event such as an endless loop occurs in a UAP, acquisition of synchronization point dumps may be skipped. If several synchronization point dumps are not acquired (instead, they are skipped), there will be an increase in the number of overwrite-disabled system log files, which may result in a shortage of system log file capacity and ultimately in abnormal termination of the unit.

This operand specifies the maximum number of times synchronization point dumps can be skipped (number of skips per transaction). When the number of skipped synchronization point dumps reaches the specified operand value, the target transaction is cancelled forcibly and rolled back. This is called the *skipped effective synchronization point dump monitoring facility*.

If Real Time SAN Replication based on the log-only synchronous method is used, the skipped effective synchronization point dump monitoring facility cannot be used at the log application site regardless of this operand value.

For details on this facility, see the *HiRDB Version 8 System Operation Guide*.

Advantage

Specifying this operand provides a means of dealing with endless loops in UAPs.

Specification guideline

Normally, specify `0` for this operand. When `0` is specified, HiRDB computes the upper limit for the skip count. If the specification of `0` causes a problem, change the value of this operand. For the method of estimating the specification value, see the *HiRDB Version 8 System Operation Guide*.

Notes

- The monitoring provided for by the specification of this operand begins the first time a synchronization point dump is collected after HiRDB is started (including a restart).
- When this operand is specified, UAPs that are being executed in the no-log mode are also monitored. If the processing of a UAP being executed in the no-log mode is cancelled, the database cannot be automatically recovered, and thus RDAREAs are placed in the error shutdown state. Therefore, when specifying the upper limit, account also for the size of the system logs that are output from other transactions inside the server during the processing of UAPs that are executed in the no-log mode.
- The `pdload`, `pdmod`, `pdrorg`, `pdexp`, `pddbst`, `pdgetcst`, `pdrbal`, and `pdvrup` commands are not monitored by this facility.

12) pd_dfw_syncpoint_skip_limit = ***maximum-number-of-skipped-synchronization-point-dumps-resulting-from-delay-of-synchronization-point-dump-acquisition-due-to-deferred-write-processing***

~ <unsigned integer>((0-100000))

If a synchronization point occurs within the synchronization point dump acquisition interval but before deferred write processing is completed, acquisition of the synchronization point dump is skipped. This is because acquisition of the synchronization point dump is delayed by the deferred write processing and the number of update buffers output by the synchronization point exceeds the number of update buffers that can be output within the synchronization point dump acquisition interval.

If more than one synchronization point dump is skipped, there will be an increase in the number of system log files that cannot be overwritten, which may result in a shortage of system log file capacity and ultimately in abnormal termination of the unit.

This operand specifies the maximum number of times synchronization point dumps can be skipped (number of skips per transaction) due to deferred write processing.

When the number of skipped synchronization point dumps due to deferred write processing reaches the specified operand value, HiRDB determines the maximum number of update buffers in such a manner that acquisition of a synchronization point dump can be completed within the synchronization point dump acquisition interval. If the maximum number of update buffers is exceeded, HiRDB then outputs the oldest update buffer and limits the total size of update buffers at a synchronization point. This is called the *update buffer size restriction facility*.

Advantage

If a synchronization point occurs within the synchronization point dump acquisition interval but before deferred write processing has terminated, the unit terminates abnormally. This operand enables such abnormal termination of the unit to be avoided.

Specification guidelines

Normally, you will omit this operand. If you wish to prevent unit abnormal termination caused by the occurrence of a synchronization point before termination of deferred write processing within the synchronization point dump acquisition interval, specify 1.

If an acceptable number of times synchronization point dumps can be skipped can be determined in advance, such as from the size of the log information, specify that value.

Operand rules

If 0 is specified in this operand, HiRDB does not use the update buffer size restriction facility.

Notes

The following notes explain the period of effectiveness of the update buffer size restriction facility. The period of effectiveness means the interval between issuance of the `KFPH23035-I` message and issuance of the `KFPH23036-I` message.

- If the number of update buffers exceeds the maximum number of update buffers determined by HiRDB, update buffers are output after update processing has executed; this degrades the update processing throughput. You can use the following formula to obtain the maximum number of update buffers determined by HiRDB:

(synchronization point dump interval / unit value of `WRITE`[#])
x (1 - (amount of log information from the previous synchronization point dump to the pre-synchronization point)
x (number of buffer sectors in buffer pool / total number of buffer sectors in buffer pool that was updated at synchronization point)

  #: For details about the unit value of `WRITE`, see the *HiRDB Version 8 System Operation Guide*.

- If the deferred write trigger is specified in the `pd_dbbuff_rate_updpage` or `pdbuffer -y` operand, and each operand value becomes greater than the maximum number of update buffers determined by HiRDB, the maximum number of update buffers determined by HiRDB is changed to the number of update buffers that triggers deferred write processing.

  The value of the `pdbuffer -w` operand is adjusted automatically so that up to the maximum number of update buffers is output for each buffer.

- A skipped synchronization point dump is detected during update buffer output processing at a synchronization point. Therefore, the update buffer size restriction facility may be enabled after a synchronization point dump is skipped and an error message is displayed.

- Normally, there is one output request per synchronization point for the number of processes that used the facility for parallel writes in deferred write processing during synchronization point processing (`pd_dfw_awt_process`).

  However, if the update buffer size restriction facility is used, there would be more than one output request in order to facilitate early detection of skipped synchronization point dumps.

8.3.4 Operands related to lock

13) pd_lck_pool_size = *server-lock-pool-size*

~ **<unsigned integer> (KB)**

- 32-bit mode: **((1-2000000))**
- 64-bit mode: **((1-2000000000))**

Specifies in kilobytes the size of the shared memory to be used by the back-end server for locking.

Specification guidelines

- In the 32-bit mode, the amount of lock pool space required for six lock requests is 1 KB.
- In the 64-bit mode, the amount of lock pool space required for four lock requests is 1 KB.
- The following formulas can be used to determine the value to be specified for this operand:

HiRDB type	Formula
HiRDB/Parallel Server (32-bit mode)	$\uparrow b \div 6 \uparrow$ (KB)
HiRDB/Parallel Server (64-bit mode)	$\uparrow b \div 4 \uparrow$ (KB)

b:

Total number of transaction lock requests to be executed concurrently by the back-end server. The number of lock requests depends on the SQL. For details about how to determine the total number of lock requests, see *E. Determining the Number of Locked Resources*.

Tuning the specified value

See the usage rate for the locked resources management table (`% OF USE LOCK TABLE`) displayed for the back-end server in the statistical information on system operation by the statistics analysis utility. If the maximum usage rate equals or exceeds 80%, it is recommended that the operand's value be increased in preparation for future database expansion. If the maximum usage rate does not exceed 10%, it is recommended that the operand's value be decreased to conserve shared memory space.

Note

If the value specified for this operand is too small, an SQL may return an error.

14) pd_lck_until_disconnect_cnt = *total-number-of-tables-and-RDAREAs-to-be-locked-until-disconnect-specification*

~ <unsigned integer>((0-140000))

Specifies the number of tables and RDAREAs to be held across transactions and the number of resources to be locked for inner replica configuration management and replica group configuration management. According to the value specified for this operand, tables, RDAREAs, and the blocks for which lock with `UNTIL DISCONNECT` is specified in inner replica configuration management and replica group configuration management are allocated in the shared memory.

Specification guidelines

Normally, this operand need not be specified. Specification of a value other than the default value may be necessary in the following cases:

- When the number of utilities to be executed concurrently increases
- When a holdable cursor is used
- When the local buffer specified in the `pdlbuffer` operand is used
- When a shared RDAREA is used

For the method of estimating the specification value for this operand, see *D.4 Formula for determining total number of tables and RDAREAs per server locked with UNTIL DISCONNECT specified (pd_lck_until_disconnect_cnt)*.

Tuning the specified value

If the value specified for this operand is small, a transaction may roll back or a utility may terminate abnormally with return code `8`. In such cases, the message `KFPA11914-E` or `KFPH28001-E` is output. If this occurs, the value of this operand should be increased.

When the value of this operand is increased, the amount of required memory space increases proportionately. The required memory size can be expressed as follows: *value-of-this-operand* × 48 (64 in the 64-bit mode) bytes.

15) pd_max_open_holdable_cursors = *maximum-number-of-holdable-cursors-that-can-be-concurrently-open-when-LOCK-statement-with-UNTIL-DISCONNECT-specification-is-not-executed*

~ <unsigned integer>((16-1024))

When you use holdable cursors for a table for which a `LOCK` statement with `UNTIL DISCONNECT` specification is not executed, this operand specifies the maximum number of holdable cursors that can be concurrently open for each transaction.

Note

Specifying a value other than the default value for this operand increases the amount of shared memory used.

Relationship to other operands

The values specified for this operand and the following operands are used for computing the shared memory size for lock servers. For a 32-bit mode HiRDB system, if the values specified for these operands are too large, the shared memory size of the lock servers exceeds 2 GB, and as a result, HiRDB may not start. Therefore, adjust the values specified for these operands so that the shared memory size of the lock servers does not exceed 2 GB.

- `pd_max_access_tables`
- `pd_max_users`
- `pd_max_bes_process`
- `pd_lck_hash_entry`
- `pd_lck_pool_size`

For details about shared memory, see the *HiRDB Version 8 Installation and Design Guide*.

16) pd_lck_hash_entry = *lock-pool-hash-entry-count*

~ <unsigned integer> ((0-2147483647))

Specifies the number of hash table entries to be used in the lock pool. According to the value specified here, HiRDB allocates a lock pool in the shared memory for the unit controller.

Specification guidelines

1. Normally, omit this operand.
2. Consider specifying a value for this operand in the following cases:
   - If you do not wish to change the shared memory size if possible when upgrading to Version 06-02 or a newer version, specify `11261`. In this case, the same number of hash entries is allocated as in the older version, and the hash table size inside the lock pool remains the same as before.
   - Specifying for this operand a value greater than the recommended value may improve performance. However, specifying a value greater than variable *a* described in *Determining the recommended value* does not improve performance over a case in which *a* is specified.

Operand rules

- If `0` is specified for this operand or if this operand and the `pd_lck_hash_entry` operand of each server definition are omitted, HiRDB computes a recommended value for each server. For details about recommended values, see *Determining the recommended value* for the `pd_lck_hash_entry` operand in the server common definition.

- If a non-zero value or a non-prime number is specified for this operand, HiRDB assumes that the largest prime number not exceeding the specified value has been specified.

Note

If the value specified for this operand is too small, hash entry shortage may occur, resulting in performance degradation. If this operand is omitted, neither hash entry shortage nor performance degradation due to hash entry shortage occurs.

Determining the recommended value

For the recommended values, see *Determining the recommended value* for the `pd_lck_hash_entry` operand of the server common definition.

17) pd_dbsync_lck_release_count = ***global-buffer-lock-release-interval-during-synchronization-point-processing***

～ <unsigned integer>((0, 100-1073741824))

Specifies an interval for unlocking global buffers, when global buffer locking occurs during synchronization point processing.

During synchronization point processing, search processing occurs on the buffers (update buffers) that must be applied to the disk. Normally, global buffers are unlocked at a specific interval during search processing on the update buffers.

For example, if 100 is specified in this operand, a global buffer is unlocked once when search processing on 100 sectors (global buffer sectors) is completed. After that, the global buffer is locked again and search processing is resumed. In this example, unlocking occurs once every 100 sectors.

Advantage

By specifying this operand, you can adjust the global buffer lock time during synchronization point processing. When a small value is specified in this operand, the global buffer lock time becomes short and transaction performance may improve during synchronization point processing.

To obtain the global buffer pool lock time, execute the statistics analysis utility and in the global buffer pool statistical information check the item called `Buffer pool lock time during synchronization point processing (SYNCL).`

Specification guidelines

Normally, there is no need to specify this operand. Consider specifying this operand when both the following conditions apply:

- Transaction performance drops during synchronization point processing.
- A large number of buffer sectors is specified in the `-n` option of the `pdbuffer` operand.

Operand rules

- If the specified value is in the range 1 to 99, 100 is set automatically.
- If 0 is specified, global buffers are locked until update buffer search processing is completed.

Notes

If a small value is specified in this operand, the update buffer search time becomes longer due to interruptions by other transactions. The global buffers updated during that time are also output during synchronization point processing. Therefore, the number of update buffers to be output during synchronization point processing increases. To obtain the number of update buffers to be output during synchronization point processing, execute the statistics analysis utility and in the global buffer pool statistical information check the item called `Number of synchronization point output pages (SYNCW).`

8.3.5 Operands related to buffers

18) pd_sql_object_cache_size = *SQL-object-buffer-size*

～ **<unsigned integer> (KB)**

- 32-bit mode: **((22-256000))**
- 64-bit mode: **((22-2000000))**

Specifies in kilobytes the size of the buffer area (shared memory) in which SQL objects are to be placed.

Specification guidelines

- SQL objects are saved in a buffer until the user's transaction has terminated. The buffer must be large enough to store the SQL objects of all transactions that will be executed concurrently.
- SQL analysis can be reduced by saving the SQL objects of static SQLs in the buffer after transaction termination (until the buffer runs out of space) and sharing them among multiple users who execute the same

UAP. To effectively utilize the buffer, it should be allocated so that the SQL objects of frequently-used UAPs are resident in the buffer.

- To estimate the buffer size, first determine the buffer size needed for UAP execution from the length of the SQL objects from the SQL statements to be issued by the UAP. Then, compute the buffer size by considering the number of UAPs that will be executed concurrently and the number of concurrently executing users.
- For details about how to estimate the length of the SQL object from a single SQL statement, see *D.2 Formulas for determining size of SQL object buffer (pd_sql_object_cache_size).*

Tuning the specified value

For details about how to tune the SQL object buffer size, see the *HiRDB Version 8 System Operation Guide*.

8.3.6 Operands related to shared memory

19) pd_bes_shmpool_size = ***back-end-server-shared-memory-size***

~ **<unsigned integer> (KB)**

- 32-bit mode: **((1-200000))**
- 64-bit mode: **((1-4000000000))**

Specifies the size of the area (units: KB) to be used by a back-end server as part of the shared memory for the unit controller.

Specification guidelines

Normally, omit this operand. HiRDB computes a value for this operand if the `pd_bes_shmpool_size` operand of the server common definition and back-end server definition is omitted. HiRDB computes this value based on *Formula for computing the shared memory used by a single server* in the manual *HiRDB Version 8 Installation and Design Guide*. If you change the value of the operand in the explanation of the variables in this formula, HiRDB automatically re-computes the value for the `pd_bes_shmpool_size` operand. Note that the value of the `pd_assurance_index_no` operand is assumed for the *total-number-of-indexes-inside-server* variable inside the formula. In the formula, for the variables *Number of global buffer pools for index* and *Total number of global buffers (number of pdbuffer operands)*, 500 is assumed in the 32-bit mode and 1000 is assumed in the 64-bit mode.

Tuning the specified value

If any of the following messages is output, increase the specification value of this operand.

- KFPA20003-E
- KFPD00005-E
- KFPD00012-E
- KFPD00021-E
- KFPH20003-E

Notes

- If an unnecessarily large value is specified for this operand, too large a shared memory area will be allocated. If the value specified for this operand is too small, the following may occur:
  - Unit will not activate
  - A UAP or utility will not execute
- If you omit this operand, make sure that you specify an appropriate value in the pd_assurance_table_no and pd_assurance_index_no operands.

8.3.7 Operands related to the RPC trace information

20) pd_rpc_trace = Y | N

Specifies whether or not RPC trace information is to be collected. HiRDB maintenance information is output in the RPC trace information.

Normally, this operand should be omitted.

Y: Collect RPC trace information.

N: Do not collect RPC trace information.

Note

Specifying Y for this operand degrades communication performance.

21) pd_rpc_trace_name = "*name-of-RPC-trace-collection-file*"

~ <pathname of up to 254 characters>

Specifies an absolute pathname for the filename for the RPC trace files. Three RPC trace files are created, with 1, 2, and *l* suffixed to the specified file name.

Note

Files with a maximum size of pd_rpc_trace_size *value* × 2 are created under the directory specified by this operand. Attention should be paid to the file capacity.

22) pd_rpc_trace_size = *RPC-trace-collection-file-size*

～ <unsigned integer> ((1024-2147483648)) (Bytes)

Specifies in bytes the size of the RPC trace files.

The value specified by this operand determines the size of both RPC trace file 1 and RPC trace file 2.

Specification guidelines

An RPC trace collects in a single file in time order the information on the communications among all processes. If the volume of this information exceeds the specified value, trace data output is switched to the other file, which means that older trace information will be overwritten. When this happens, the amount of trace information that is available may be inadequate, making troubleshooting difficult. For this reason, at least 1,000,000 should be specified for this operand.

Note

The value specified by this operand does not apply to RPC trace file *l*, because its size is fixed at 0 bytes.

8.3.8 Operands related to troubleshooting information

23) pd_module_trace_max = ***maximum-number-of-module-traces-that-can-be-stored***

～ <unsigned integer>((126-16383))

A HiRDB process records the history of the executed functions and macros inside the process private memory. This history is called a *module trace*. This operand specifies the number of module trace records. The content of this history is loaded into the `core` file and is output when a process error occurs.

Specification guidelines

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

Process private memory of the following size is allocated to each process:

In the 32-bit mode: 64 + 48 × *value of* `pd_module_trace_max` operand (bytes)

In the 64-bit mode: 64 + 64 × *value of* `pd_module_trace_max` operand (bytes)

24) pd_module_trace_timer_level = 0 | 10 | 20

Specifies how to acquire the time to be output in module traces. The following table explains the meaning of the value specified for this operand.

Specified value	Time acquisition method
0	Time is output in seconds at every module trace output location.
10	Time is output in microseconds only at performance-critical module trace output locations, such as those before and after input/output processing, and time is output in seconds at other locations.
20	Time is output in microseconds at every module trace output location.

Specification guidelines

Normally, there is no need to specify this operand. If a maintenance engineer asks you to specify this operand for a performance check purpose or the like, follow the maintenance engineer's instructions.

Note

If you specify a value other than 0 for this operand, a function for acquiring time in microseconds is issued, and as a result, system performance may decline.

8.3.9 Operands related to global buffers

25) pd_max_add_dbbuff_no = ***maximum-global-buffers-count-for-dynamic-addition***

~ <unsigned integer>((1-32752))

In order to change global buffers dynamically, this operand specifies the maximum number of global buffers (per server) that can be added dynamically by the pdbufmod command.

Condition

Y must be specified in the pd_dbbuff_modify operand.

Specification guidelines

- Estimate the number of global buffers to be added dynamically by the pdbufmod command and then specify a sufficient value based on that value.
- Determine the operand's value in such a manner that the following condition is satisfied:

  Value of pd_max_add_dbbuff_no ≤ 2000000 - number of global buffers allocated per server during HiRDB startup

Notes

Do not specify an unnecessarily large value in this operand. If this operand's value is too large, the shared memory used by HiRDB increases, which may result in a shortage of shared memory and an inability of HiRDB to start.

Relationship to other operands

This operand is related to the following operands:

- SHMMAX
- pdbuffer
- pd_max_add_dbbuff_shm_no

26) pd_max_add_dbbuff_shm_no = *maximum-shared-memory-segments-count-for-dynamic-addition*

~ <unsigned integer>((1-32752))

In order to change global buffers dynamically, this operand specifies the maximum number of shared memory segments (per server) that can be allocated when dynamic addition is performed by the pdbufmod command.

Condition

Y must be specified in the pd_dbbuff_modify operand.

Specification guidelines

Estimate the number of global buffers to be added dynamically by the pdbufmod command and then specify an appropriate value.

Notes

- If the following condition is satisfied, the value of the pd_max_add_dbbuff_no operand is assumed in this operand:

  Value of pd_max_add_dbbuff_shm_no < value of pd_max_add_dbbuff_no

  The value of the pd_max_add_dbbuff_no operand is also assumed when the default value satisfies the above condition.
- Do not specify an unnecessarily large value in this operand. If this operand's value is too large, the shared memory used by HiRDB increases, which may result in a shortage of shared memory and an inability of HiRDB to start.
- If the size of a shared memory segment to be added exceeds the SHMMAX operand value, shared memory is divided into multiple segments based on the SHMMAX operand value as the maximum value. Either increase the SHMMAX operand value based on the size of the shared memory segment to be added or increase the pd_max_add_dbbuff_no

operand value so that no shortage occurs when the shared memory is segmented.

- If you have changed the value of this operand, re-evaluate the maximum size of a shared memory segment in the OS parameters, the maximum number of shared memory segments in the system, and the maximum number of shared memory segments per process.

Relationship to other operands

This operand is related to the following operands:

- `SHMMAX`
- `pdbuffer`
- `pd_max_add_dbbuff_no`

8.3.10 Operands related to delayed batch creation of plug-in index

27) pd_plugin_ixmk_dir = "*index-information-file-creation-directory-name*" or "*index-information-file-creation-HiRDB-file-system-area-name*"

~ <pathname>

Specifies the name of the directory under which the index information file for delayed batch creation of a plug-in index is to be created. Specify a HiRDB file system area name in order to create the index information file in a HiRDB file system area. An absolute pathname must be used for the directory name or HiRDB file system area name.

For details about delayed batch creation of a plug-in index, see the *HiRDB Version 8 System Operation Guide*.

Notes

- The directory (or HiRDB file system area) specified here must have been created in advance. If a nonexistent directory (or HiRDB file system area) is specified, an error will result during execution of a UAP that specifies delayed batch creation of plug-in indexes (UAP that is executed in an environment in which `PDPLGIXMK = YES` is specified in the client environment definition).
- Once the UAP has executed, the value specified for this operand must not be changed before delayed batch creation of plug-in indexes is performed by the database reorganization utility. If it is changed, a plug-in index delayed batch creation cannot be performed.

8.3.11 Operands related to Java

Java operands are specified when a Java stored procedure or Java stored function is used. For details about Java stored procedures and Java stored functions, see the

HiRDB Version 8 UAP Development Guide.

Note that some platforms on which HiRDB is run do not support Java stored procedures and Java stored functions. For details, see *Environment supporting Java stored procedures and functions* in the *HiRDB Version 8 System Operation Guide*.

28) pd_java_stdout_file = *"Java-virtual-machine-standard-output-and-standard-error-output-destination-file"*

~ **<pathname>**

Specifies as an absolute pathname the file to which the standard output and standard error output are to be output in a Java virtual machine. If this operand is omitted, the standard output and standard error output of the Java virtual machine are ignored.

Specification guideline

Because the size of the file specified by this operand is extremely large, this operand is not normally specified. It is recommended that this operand be specified during debugging of a Java stored procedure or Java stored function. There is no limit to the size of the file that can be specified by this operand.

Note

If there are simultaneous writing attempts from multiple processes, their output contents cannot be guaranteed.

Operand rules

- Up to 255 characters can be used for the pathname.

8.3.12 Operands related to HiRDB External Data Access facility

For details on the HiRDB External Data Access facility, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

Note that some platforms on which HiRDB is run do not support the HiRDB External Data Access facility. For details, see *HiRDB External Data Access* in the manual *HiRDB Version 8 Description*.

29) pd_foreign_server_libpath = ***foreign-server-client-library-path-name*** **[,"*foreign-server-client-library-path-name*"]...**

~ **<pathname>**

Specifies an absolute path name for a foreign server client library. For the path name of a foreign server client library, see the documentation for the DBMS of the foreign server to be connected.

Condition

HiRDB External Data Access is required.

Operand rule

The maximum number of characters that can be specified is 512.

Note

Note the following when using the standby-less system switchover (1:1) facility: If foreign servers are defined for both a normal BES and an alternate BES, and if the server type (DBMS to be accessed) is the same for these foreign servers, set the same version for the foreign server client libraries.

8.3.13 Operands related to system log files

30) pd_log_dual = Y | N

Specifies whether or not dual system log files are to be used.

`Y`: Use dual system log files.

`N`: Do not use dual system log files.

Advantages

When dual system log files are used, HiRDB collects the same system log information in both files, which are called File A and File B. If a failure occurs in one of the files while the collected system log file is being loaded, the file can be loaded from the other file, resulting in higher system reliability.

Relationship to other operands

When use of dual system log files is specified, the name of the File B system log file must be specified with the `pdlogadpf` operand.

31) pd_log_dual_write_method = serial | parallel

This operand is applicable only to the AIX 5L version of HiRDB.

This operand specifies whether to use the facility for parallel output of system logs. For details about this facility, see the *HiRDB Version 8 Installation and Design Guide*.

`serial`: The facility for parallel output of system logs is not used.

`parallel`: The facility for parallel output of system logs is used.

When `parallel` is specified for this operand, HiRDB uses the Asynchronous I/O Subsystem to execute parallel output to the system log files.

Conditions

When you specify `parallel` for this operand, the following conditions must be satisfied:

- `pd_log_dual = Y`
- The system log files are located in character special files.

If either of these conditions is not satisfied, the facility for parallel output of system logs is not used regardless of the value specified for this operand.

Additionally, when you specify `parallel` for this operand, the Asynchronous I/O Subsystem must be installed.

32) pd_log_remain_space_check = warn | safe

Specifies the processing to be performed by HiRDB when the available space in the system log file falls below the warning level. This is called the facility for monitoring the free area for the system log file. For details on this facility, see the *HiRDB Version 8 System Operation Guide*.

`warn`:

When the available space in the system log file falls below the warning level, the `KFPS01162-W` message is output.

`safe`:

When the available space in the system log file falls below the warning level, scheduling of new transactions is suppressed, all transactions inside the server are forcibly terminated, and the `KFPS01160-E` message is output.

Specification guideline

Hitachi recommends the specification of `safe` because it can reduce the probability of abnormal termination of units due to system log file space shortage. However, when `safe` is specified, all transactions inside the server are forcibly terminated when a shortage occurs in the available space in the system log file. Therefore, the design of the system log file requires more accuracy. For details on system log file design, see the *HiRDB Version 8 Installation and Design Guide*.

33) pd_log_auto_unload_path = "*unload-log-file-output-directory*"[,"*unload-log-file-output-directory*"]...

~ <pathname>((1-136 characters))

Specifies as absolute pathnames the unload file output directories when the automatic log unloading facility is to be used for the system log. A HiRDB file system area name must be specified to create the unload log file in a HiRDB file system area. *The directories or HiRDB file system areas specified for this operand must be created before HiRDB is started.*

Additionally, in this operand, specify a different directory or HiRDB file system area for each server.

For details about the automatic log unloading facility, see the *HiRDB Version 8 System Operation Guide*.

Specification guidelines

It is important to check the available disk space before specifying a directory, so as to ensure that the created unload log file does not cause a disk space shortage.

If an unload log file cannot be created in the specified directory because of a disk space shortage, the automatic log unloading facility stops. If this is a possibility, creation of multiple directories is recommended.

Note, however, that the database recovery operation of selecting the unload files needed for recovery is simplified somewhat when only one directory is used.

Also keep in mind the following when multiple directories are created:

- It is recommended that directories be specified in different partitions to protect against disk errors.
- If the unload log file cannot be created in a single directory because of a full disk or disk error, create an unload log file under a different directory. HiRDB uses the directories specified by this operand in the order of their specification.

Operand rules

- Up to 128 directories can be specified.
- When multiple directories are specified, the same pathname cannot be specified.

Notes

The automatic log unloading facility cannot be used in the following cases:

- `N` is specified in the `pd_log_unload_check` operand.
- The log application site uses Real Time SAN Replication based on the log-only synchronous method (the `KFPS04689-W` message is displayed).

34) pd_log_singleoperation = Y | N

This operand is applicable when dual system log files are used; it need not be specified when dual system log files are not used.

Specifies whether or not the single-operation mode is to be used for the system log files. Even if an error occurs in a system log file, making no dual system log files available, HiRDB (or a unit for a HiRDB/Parallel Server) can continue processing using the remaining single normal system log file without being abnormally terminated. This is called *single operation of the system log files*.

The mode in which continuation of processing is permitted only with both system log files available (normal operation mode) is called *double operation of the system log files*.

`Y`: Use single operation of the system log files.

`N`: Do not use single operation of the system log files. Both system log files must always be used.

Condition

This operand is valid only when `pd_log_dual = Y` is specified.

35) pd_log_rerun_reserved_file_open = Y | N

Specifies whether or not a system log file is to be opened automatically.

If no system log file that can be overwritten is available during a unit restart, HiRDB opens a reserved file (if one is available), makes it overwritable, and continues processing. This is called *automatic opening of system log file*.

A reserved file is used in the following cases:

- Between the time of a restart and the time when the first synchronization point dump is collected
- When none of the opened file groups can be overwritten

`Y`: Open a system log file automatically (open and use a reserved file).

`N`: Do not open a system log file automatically (do not use a reserved file).

Advantages

When `Y` is specified, the unit can be restarted as long as a reserved file is available, even though no file that can be swapped in is available during the unit restart.

However, if a file that is in unload wait status is available, the unit is stopped; once that file has been unloaded, the unit can be restarted.

36) pd_log_rerun_swap = Y | N

Specifies whether or not the system log files are to be swapped during a unit restart.

`Y`: Swap the system log files.

`N`: Do not swap the system log files.

Advantage

When Y is specified, there can be a physical separation between the system log files used before and after a restart. Therefore, the system log file that was being used before the restart can be reused during server operation.

37) pd_log_swap_timeout = *wait-time-for-completion-of-system-log-file-swapping*

～ <unsigned integer>((1-32580)) (seconds)

Specifies in seconds the wait time during which swapping of system log files must be completed. If a swap of system log files is not completed within the specified amount of time, the unit terminates abnormally.

Specification guidelines

Normally, there is no need to specify this operand. If it takes a long time to swap system log files for a reason such as poor machine performance, you can specify this operand with a value that is greater than the default value. To detect errors or delays quickly during system log file swapping so that the unit can be terminated (such as because of a disk failure), reduce the value of this operand.

38) pd_log_unload_check = Y | N

Specifies whether or not HiRDB is to check the unload status of system log files.

Y:

Check the unload status (normal operation).

N:

Do not check the unload status. A system log file is placed in swappable status, regardless of its unload status, when both of the following conditions are satisfied:

- It is in overwritable status
- It is in extraction completed status (HiRDB Datareplicator)
- Overwriting permitted status for online reorganization (HiRDB Staticizer Option)

In such a case, the system log file operation method is to release checking of unload status. For details about this operation method, see the *HiRDB Version 8 System Operation Guide*.

Advantages

Specifying N provides the following advantages:

- Operations are simplified because the system log file unload operation is eliminated.

- It is not necessary to provide files for storing unload files.

Specification guideline

`N` should be specified if the system log file will not be needed for database recovery (i.e., if recovery from a backup collection point will be sufficient).

Notes

The following points apply when `N` is specified:

- Database can be recovered only if backups have been made.
- If this option is specified when the system log file is required for database recovery, it will not be possible to recover the database.

39) pd_log_max_data_size = *log-input/output-buffer-size*

～ <unsigned integer>((32000-523000)) (Bytes)

Specifies in bytes the size of the buffer to be used for system log input/output operations.

Specification guideline

Specify a value that satisfies conditional expression 1 below. If `uap` is specified in the `pd_rpl_reflect_mode` operand or a recovery-unnecessary front-end server is used, specify a value that satisfies both conditional expressions below (1 and 2). To optimize the value, use the tuning method for the specification value.

Conditional expression 1: log input/output buffer length ≥ *a*

a: 72 x total number of back-end servers + 1432

Conditional expression 2: log input/output buffer length ≥ *b*

b: (Maximum number of back-end servers subject to update processing by a single transaction + 1) x 128 + 64

Tuning the specified value

A value (other than the default value) may need to be specified for this operand after the following types of statistics analysis utility statistical information related to system operation have been checked:

- Number of buffer sectors waiting for input/output (`# OF BUFFER FOR WAIT I/O`)

  If the average number of buffer sectors waiting for input/output significantly exceeds 100, increase the value for this operand so that the average approaches 100.

- Number of waits caused by lack of a current buffer (`# OF WAIT`

THREAD)

If the number of waits caused by lack of a current buffer is not 0, increase the value for this operand.

Relationship to other operands

Use this operand and the pd_log_write_buff_count operand to determine the log output buffer size.

40) pd_log_write_buff_count = *log-output-buffer-sectors-count*

~ <unsigned integer>((3-65000))

Specifies the number of buffer sectors to be used for system log output.

Tuning the specified value

A value (other than the default value) may need to be specified for this operand after the following types of statistics analysis utility statistical information related to system operation have been checked:

- Number of times the buffer was full (# OF BUFFER FULL)
- Number of waits caused by lack of a current buffer (# OF WAIT THREAD)

If either of these values is large, a larger value should be specified in order to improve throughput.

Relationship to other operands

Use this operand and the pd_log_max_data_size operand to determine the log output buffer size.

41) pd_log_rec_leng = *system-log-file-record-length*

~ <unsigned integer>((1024, 2048, 4096)) (Bytes)

Specifies the record length for the system log files; the specifiable values are 1024, 2048, and 4096.

The record length specified in the -l option of the pdloginit command should be specified for this operand.

Notes

- If a value that is different from the record length specified by the -l option of the pdloginit command is specified for this operand, system log files cannot be opened.
- For details about how to modify the system log file record length, see the *HiRDB Version 8 System Operation Guide*.

8.3.14 Operands related to synchronization point dump files

42) pd_spd_dual = Y | N

Specifies whether to use dual synchronization point dump files.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for the server are also used at the log application site.

`Y`: Uses dual synchronization point dump files.

`N`: Does not use dual synchronization point dump files.

Advantage

When dual synchronization point dump files are used, HiRDB collects the same synchronization point dump in both dump files A and B. Even when an error occurs in one of the files when the collected synchronization point dump is being loaded, the synchronization point dump can be loaded from the other file, thus improving system reliability.

Relationship to other operands

To use dual synchronization point dump files, specify the name of synchronization point dump file B in the `pdlogadpf` operand.

43) pd_spd_assurance_msg = Y | N

Specifies whether or not the `KFPS02183-I` message is to be output when a synchronization point dump is completed.

`Y`: Output the message.

`N`: Do not output the message.

44) pd_spd_assurance_count = *number-of-guaranteed-valid-generations*

~ <unsigned integer>((1-2))

Specifies the range of system log files to be saved during HiRDB operation as a protection against events such as a synchronization point dump file input error during system recovery. What is specified here is a number of synchronization point dump file generations; the specified value is referred to as the *number of guaranteed-valid generations*. The number of past generations of synchronization point dump files specified here are overwrite disabled.

If Real Time SAN Replication based on the log-only synchronous method is used, this operand is ignored, if specified, at the log application site. The system log files used at the log application site are overwritten by the transaction execution site even if they include system log information that has not been applied.

Advantage

When 2 is specified as the number of guaranteed-valid generations and an error occurs in the most recent synchronization point dump file generation, the system can be recovered using the preceding synchronization point dump file generation, resulting in higher reliability.

Specification guideline

- To improve reliability, 2 should be specified for the number of guaranteed valid generations. However, the number of overwrite disabled synchronization point dump files increases (to two).
- If reliability has been improved with the use of dual synchronization point dump files, Hitachi recommends that you omit this operand or specify 1 for it.

Notes

- The minimum number of synchronization point dump files needed is *number-of-guaranteed-valid-generations* + 1.
- Specifying 2 increases the number of overwrite disabled synchronization point dump files. Moreover, the system log files that correspond to the overwrite disabled synchronization point dump files are also overwrite disabled. Consequently, specifying 2 for the number of guaranteed valid generations increases the number of overwrite disabled system log files. As a result, a shortage may occur in the number of system log files that can be swapped in. To prevent this, it may be necessary to reevaluate the system log file capacity.

45) pd_spd_reduced_mode = *reduced-mode-operation-option*

～ <unsigned integer>((0-2))

Specifies whether or not the reduced mode operation for synchronization point dump files is to be used.

Reduced mode operation is a facility for continuing processing if at least two synchronization point dump files can be used, even if an event such as a file error during HiRDB operation or restart reduces the number of synchronization point dump files to or below the number of required files (*number of guaranteed valid generations*[*] + 1).

[*] Value specified for the pd_spd_assurance_count operand.

0: Do not use the reduced mode operation.

1: Use the reduced mode operation.

2: Use the reduced mode operation and issue a warning message whenever a synchronization point dump is being acquired during the reduced mode operation.

46) pd_spd_reserved_file_auto_open = Y | N

Specifies whether or not a synchronization point dump file is to be opened automatically. When Y is specified and an error in a synchronization point dump file reduces the number of synchronization point dump files to the number of files necessary for operation (*number of guaranteed valid generations*[*] + 1), HiRDB opens a reserved file (if one is available), makes it overwritable, and continues processing. This is called *automatic opening of synchronization point dump file*.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for server are also used at the log application site.

[*] Value specified for the pd_spd_assurance_count operand.

Y:

Open a synchronization point dump file automatically. A reserved file is opened when the number of files falls below the number necessary for operation (*number of guaranteed valid generations* + 1).

N:

Do not open a synchronization point dump file. No reserved file is opened, even though the number of files falls below the number necessary for operation (*number of guaranteed valid generations* + 1).

Relationship to other operands

This operand has a higher priority than the pd_spd_reduced_mode operand.

47) pd_spd_max_data_size = *synchronization-point-dump-file-buffer-size*

～ <unsigned integer>((32000-4000000)) (Bytes)

Specifies in bytes the size of the buffer (shared memory) to be used for synchronization point dump file input/output operations.

The value specified here affects the number of synchronization point dump file input/output operations.

If Real Time SAN Replication based on the log-only synchronous method is used, the synchronization point dump files for log application processing for server are also used at the log application site.

Specification guidelines

- Normally, this operand need not be specified.
- When the value of this operand is increased, the number of synchronization point dump file input/output operations is reduced.

48) pd_log_sdinterval = *system-log-output-volume*[,interval]

Specifies the collection interval for synchronization point dumps. The following should be taken into consideration in specifying this operand:

- Volume of system log information output since the previous synchronization point
- Amount of time that has elapsed since the previous synchronization point

***system-log-output-volume*: ~ <unsigned integer>((100-100000)) (Number of log blocks)**

Specifies an interval between synchronization point dumps in terms of the number of blocks of log information. A synchronization point dump is collected each time system log information equivalent to the number of log blocks specified here has been output.

***interval*: ~ <unsigned integer>((0 or 10-1440)) (Minutes)**

Specifies a synchronization point dump collection interval in terms of number of minutes between synchronization point dumps.

- When `0` is specified for the interval, HiRDB does not use a time interval for collecting synchronization point dumps.
- If no transactions execute during an interval, no synchronization point dump is collected even though the amount of the time specified here has elapsed.

Specification guidelines

- This operand need not be specified if no specific amount of time is specified for restarting HiRDB.
- The value specified for this operand affects the amount of time required to restart HiRDB.
- Specifying a small value for this operand reduces the amount of time required for database recovery during a HiRDB restart. However, because the frequency of synchronization point dumps increases, online performance may deteriorate in some cases.

  Conversely, specifying a large value for this operand increases the amount of time required for database recovery during a HiRDB restart. However, because the frequency of synchronization point dumps decreases, online performance may improve in some cases.

Tuning the specified value

The synchronization point dump collection interval can be checked with the statistics analysis utility; the relevant information is shown under `SYNC`

POINT GET INTERVAL in the statistical information related to system operation. The average SYNC POINT GET INTERVAL value should be used. If the synchronization point dump collection interval is determined to be too long, the specification value should be decreased; conversely, if it is determined to be too short, the specification value should be increased.

Note

The synchronization point dump collection interval is determined on the basis of the volume of system log information that is output. Therefore, committing data from memory to a database takes a long time during intervals that have few updating transactions. If an error occurs at such a time, it will take longer to recover the transactions that were generated during that period. If this is a possibility, the synchronization point dump collection interval should be set by also using the interval value.

8.3.15 Operands related to server status files

49) pd_sts_file_name_1 = "*logical-file-name*", "*file-a-status-file-name*", "*file-b-status-file-name*"

: . . .

pd_sts_file_name_7 = "*logical-file-name*", "*file-a-status-file-name*", "*file-b-status-file-name*"

Define server status files. Although the pd_sts_file_name_2 to 7 operands may be omitted, the pd_sts_file_name_1 operand cannot be omitted.

"*logical-file-name*" ~ <identifier>((1-8 characters))

Specifies the logical file name of a status file for the back-end.

When a command that manipulates the status file is to be executed, the logical file name defined here must be specified.

"*file-a-status-file-name*" ~ <pathname>((up to 167 characters))

Specifies the name of the File A status file as an absolute pathname.

"*file-b-status-file-name*" ~ <pathname>((up to 167 characters))

Specifies the name of the File B status file as an absolute pathname.

Specification guidelines

- The files specified as File A and File B should be status files created with the pdstsinit command.
- If a file that has not been created with the pdstsinit command is specified, a virtual status file is created.
- If an error occurs in a status file, HiRDB swaps the status files. If no

spare file is available for swapping, HiRDB (or a unit for a HiRDB/ Parallel Server) terminates abnormally. For this reason, defining a large number of system files improves system reliability, but at the expense of increasing the amount of disk space that is required.

- The status files specified for File A and File B must have the same record length and capacity.

Operand rules

- Up to seven instances of this operand may be specified.
- A status file has a dual structure consisting of File A and File B; both must be specified.
- Environment variables cannot be used for the absolute pathnames of the File A and File B file names.
- The same names cannot be specified for the logical file name, the File A file name, and the File B file name.

Notes

- When HiRDB is started normally, the primary file (the file that was the primary file when HiRDB was terminated) is inherited. However, if there is no current file that can be inherited, for example because all status files have been initialized, the first status file that was specified among those specified by the `pd_sts_file_name_1` to `7` operands becomes the current file, and the remaining files become spare files if they can be opened. Any files that cannot be opened become reserved files.
- When HiRDB is restarted, the primary file (the file that was the primary file when HiRDB was terminated) is inherited.

Using a virtual status file

Specifying a virtual status file makes it possible to add a new status file while HiRDB is running.

For example, if the number of spare files is reduced because of errors in status files, virtual status files can be converted into spare files. The procedure for converting virtual status files into spare files follows.

Procedure

1. Use the `pdstsinit` command to create a status file in a HiRDB file system area for system files.
2. Use the `pdstsopen` command to open the status file.

These operations can be performed while HiRDB is running; there is no need to stop HiRDB.

- **Advantages and disadvantages**

  Defining a virtual status file reduces the amount of space required in the HiRDB file system area. However, a virtual status file cannot be added as a spare file if the HiRDB file system area for system files does not have sufficient space (sufficient space for adding the file), thus reducing system reliability.

  When virtual status files are not defined, the amount of space required in the HiRDB file system area is increased. However, because a swap destination is guaranteed when a file error occurs, system reliability is increased.

- **Notes**

  When virtual status files are defined, HiRDB determines that a status file error has occurred when HiRDB is started. For this reason, HiRDB cannot start if `stop` (default value) is specified for the `pd_sts_initial_error` operand. When virtual status files are defined, specify `continue` or `excontinue` for the `pd_sts_initial_error` operand. It is also necessary before starting HiRDB to specify the current file in the `pd_sts_last_active_file` operand.

Relationship to other operands

This operand is related to the `pd_sts_subfile_name_1` to `7` operands.

50) pd_sts_subfile_name_1 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"

:

pd_sts_subfile_name_7 = "*logical-file-name*","*primary-status-file-name-for-log-application-processing*","*secondary-status-file-name-for-log-application-processing*"

Defines the server status files for log application processing, which are used at the log application site when Real Time SAN Replication based on the log-only synchronous method is used.

"*logical-file-name*": ～ <identifier> ((1-8 characters))

Specifies the logical file name of a server status file for log application processing. This must be a logical file name specified in `pd_sts_file_name_1` through `pd_sts_file_name_7`. A logical file name defined here is used for executing commands that manipulate the status files for log application processing.

"*primary-status-file-name-for-log-application-processing*": ～ <path name> ((up to 167 characters)

Specifies the absolute path name of a primary status file for log application processing.

"*secondary-status-file-name-for-log-application-processing*": ～ <path name> ((up to 167 characters)

Specifies the absolute path name of a secondary status file for log application processing.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.

Specification guidelines

- For the primary and secondary status file names for log application processing, specify the names of status files for log application processing that were created in the preparations for log application. If any other status file for log application processing is specified, the specified file becomes a *virtual status file for log application processing*.
- If an error occurs in a status file for log application processing, HiRDB swaps status files for log application processing. If no spare file is available for swapping, HiRDB (or the unit in the case of a HiRDB/Parallel Server) terminates abnormally. For this reason, defining a large number of status files improves system reliability, but at the expense of increasing the amount of disk space that is required.
- The specified primary and secondary status files for log application processing must have the same record length and capacity.

Operand rules

- You can specify a maximum of 7 instances of this operand.
- This operand is ignored at the transaction execution site.
- If you performed the preparations for log application during HiRDB startup, create status files for log application processing that correspond to all status files for transaction processing that can be opened and that were specified in `pd_sts_file_name_1` through `pd_sts_file_name_7`.

- The system uses dual status files for log application processing. Make sure that you specify both files.
- Environment variables cannot be used for the absolute path names of the primary and secondary status files for log application processing.
- Each logical file name, primary status file name for log application processing, and secondary status file name for log application processing must be unique.
- The names of the primary and secondary status files for log application processing must be different from the primary and secondary status file names specified in the `pd_sts_file_name_1` through `pd_sts_file_name_7` operands.

Notes

When HiRDB is started normally or restarted, the current file (the file that was the current file when HiRDB last terminated) is inherited. However, if there is no current file that can be inherited, such as when all status files for log application processing have been initialized, the log application site can no longer be started. If this occurs, perform the preparations for log application.

How to use a virtual status file for log application processing

See the description of the `pd_sts_file_name_1` through `pd_sts_file_name_7` operands.

Relationship to other operands

This operand is related to the `pd_sts_file_name_1` through `pd_sts_file_name_7` operands.

8.3.16 Operands related to server status files (when an error occurs)

For details about the measures to be taken when an error occurs in a status file, see the *HiRDB Version 8 System Operation Guide*.

51) pd_sts_initial_error = stop | continue | excontinue

When a server (single server, front-end server, dictionary server, or back-end server) starts, HiRDB performs a process of identifying the current server status file. This operand specifies the action that HiRDB takes when any of the following errors are detected during this identification process.

- No real server status file is found.
- An error is detected in the server status file.

The current file identification process is applied to the server status files specified by the `pd_sts_file_name_1` to `7` operands.

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing specified in the pd_sts_subfile_name1 through pd_sts_subfile_name7 operands are also used at the log application site. For the log application site, replace the pd_sts_last_active_file and pd_sts_last_active_side operands in the above description with the pd_sts_last_active_subfile and pd_sts_last_active_side_sub operands, respectively.

stop:

When an error is detected in a server status file during the current file identification process, the startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started. In this case, first take a corrective action for the status file in which the error was detected, and then start HiRDB.

continue or excontinue:

Even when an error is detected in the server status file during the current file identification process, the startup of the server is continued if the current file is normal. However, the startup may be stopped depending on the value specified for the pd_sts_singleoperation operand (whether operation should continue with a single status file). The following table shows the relationship to the pd_sts_singleoperation operand.

- **Relationship to the pd_sts_singleoperation operand**

pd_sts_singleoperation operand value	Processing by HiRDB	HiRDB administrator's action
continue	When an error is detected in a server status file, HiRDB cannot identify the current file, and thus startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.	The HiRDB administrator identifies the current file and specifies the pd_sts_last_active_file and pd_sts_last_active_side operands. Afterwards, start HiRDB.
stop (default value)	When an error is detected in a server status file, HiRDB identifies the current file and startup of the server is continued. However, if the primary and secondary files satisfy any of the conditions listed in the table below (cases in which HiRDB cannot identify the current file), startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.	If HiRDB cannot identify the current file, the HiRDB administrator identifies the current file and specifies the pd_sts_last_active_file and pd_sts_last_active_side operands. Afterwards, start HiRDB.

- **When HiRDB cannot identify the current file**

pd_sts_initial_error operand value	File A status	File B status
continue	Error shutdown	Error shutdown
	Error shutdown	Open (initial state)
	Error shutdown	No real file
	Open (initial state)	Error shutdown
	Open (initial state)	No real file
	No real file	Open (initial state)
	No real file	Error shutdown
	No real file	No real file
excontinue	Error shutdown	Error shutdown
	Error shutdown	No real file
	No real file	Error shutdown
	No real file	No real file

Therefore, to minimize actions by the HiRDB administrator (to reduce the number of cases in which both the pd_sts_last_active_file and pd_sts_last_active_side operands must be specified), specify the following:

- pd_sts_initial_error = excontinue
- pd_sts_singleoperation = stop

Specification guidelines

The following table shows the specification guidelines, along with their advantages and disadvantages.

Item	pd_sts_initial_error operand value	
	stop	continue or excontinue
Processing by HiRDB during server startup	When an error is detected in a server status file, startup of the server is stopped and HiRDB (or the unit in the case of a HiRDB/ Parallel Server) is not started.	Even when an error is detected in a server status file, the startup of the server is continued if the current file is normal.

Item	pd_sts_initial_error operand value	
	stop	continue or excontinue
Specification guideline	To improve system reliability, specify `stop`.	To simplify the error-handling actions during HiRDB startup, specify `continue` or `excontinue`.
Advantage	Guarantees that all server status files of the server are normal when the server starts. Therefore, if an error occurs in the current file after HiRDB has started, it can be swapped to a spare file.	Even when an error is detected in a server status file during server startup, HiRDB can start with the remaining normal files only. Therefore, HiRDB stop time can be shortened. In this case, because the number of spare files has become small, it is necessary to immediately repair the status files containing errors.
Disadvantage	Possibility increases that an error in a server status file stops the startup of HiRDB.	Because HiRDB may be running with only a small number of spare files, system reliability is low. Depending on the number of spare files available, it may not be possible to swap server status files.

Notes

- If both current files are abnormal, startup of the server is stopped regardless of the value specified for this operand, and HiRDB (or the unit in the case of a HiRDB/Parallel Server) is not started.
- Before starting HiRDB, do not initialize the current file with the `pdstsinit` command. If the current file is initialized, HiRDB cannot be restarted.

Remarks

For details about the operand value, HiRDB processing, and HiRDB administrator's action, see *Remarks* in the section on the `pd_sts_initial_error` operand.

52) pd_sts_singleoperation = stop | continue

Specifies whether or not processing may continue in the single-operation mode for server status files.

The status file single-operation mode means that processing is to continue using only the normal file (single file) when an error occurs in the status file and a spare file is not available. For details about the status file single-operation mode, see the *HiRDB Version 8 Installation and Design Guide*.

If an error occurs in one of the current files and a spare file is available, the status file is swapped and processing continues regardless of the value specified for this operand (operation in the single-operation mode does not occur).

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing are also used at the log application site.

`stop`:

> Do not permit operation in the single-operation mode. If operation in the single-operation mode is necessary, HiRDB (or a unit for a HiRDB/Parallel Server) is abnormally terminated. If HiRDB is abnormally terminated, allocate spare files and then restart HiRDB.

`continue`:

> Enable operation in the single-operation mode. When the single-operation mode goes into effect, the message `KFPS01044-I` is output. If an error occurs in the normal file during operation in the single-operation mode, or if HiRDB (or a unit for a HiRDB/Parallel Server) is abnormally terminated while the status file is being updated, HiRDB cannot be restarted. Therefore, when the single-operation mode goes into effect, immediately allocate spare files.

Specification guidelines

- Hitachi recommends that you specify `stop` to increase system reliability. Hitachi also recommends that you increase the number of spare files to guard against errors in the current files.
- The following table shows the specification guidelines, along with their advantages and disadvantages.

Item	pd_sts_singleoperation operand value	
	stop	continue
Specification guideline	To improve system reliability, specify `stop`.	Specify `continue` if it is important not to stop HiRDB.
Advantage	When an error occurs in one of the current files and a spare file is not available, operation in the single-operation mode does not occur and HiRDB is abnormally terminated. Consequently, the possibility of losing the content of the current files is reduced.	Even when an error occurs in one of the current files and a spare file is not available, processing can be continued. Therefore, the possibility that an error in the status file stops HiRDB is reduced.

Item	pd_sts_singleoperation operand value	
	stop	continue
Disadvantage	Possibility that an error in the status file stops HiRDB increases. However, increasing the number of spare files can reduce this possibility.	If an error occurs in the normal status file during operation in the single-operation mode or if HiRDB is abnormally terminated during updating of the status file, the content of the current file is lost and HiRDB cannot be restarted.

Relationship to other operands

The combination of the values specified for the `pd_sts_singleoperation` and `pd_sts_initial_error` operands determines the processing by HiRDB when an error occurs in the status file. Therefore, the values to be specified for these operands should be determined together.

53) pd_sts_last_active_file = "*logical-file-name*"

~ <identifier> ((1-8 characters))

Specifies the name of the logical file to be used as the current status file at the time of the back-end server startup. HiRDB compares the file name specified in this operand with the file name selected by HiRDB to be the current file. If the file names match, HiRDB (or a unit for a HiRDB/Parallel Server) is started; otherwise, HiRDB (or a unit for a HiRDB/Parallel Server) is not started.

Conditions

The following conditions must be satisfied:

- `continue` or `excontinue` is specified in the `pd_sts_initial_error` operand.
- It cannot be determined if the current file selected by the HiRDB system was the most recent current file during the previous session.

Specification guidelines

1. To start HiRDB immediately after initializing all status files:

   From among the operable logical files specified in the `pd_sts_file_name1` to `7` operands, specify the one that has the smallest number. Forced startup will be used in this case, regardless of the previous termination mode.

2. When both of the current files are normal:

   Specify the name of the current file.* If HiRDB cannot be started even though the name of the current file is specified, the current file may have been initialized. In this case, first initialize all status files, then use the method in 1 above to start HiRDB (forced startup will be used, regardless of the previous termination mode).

3. When one of the current status files that were active during the previous operation has an error

   Use the method in 2 previously, with the following operands specified:

   - `pd_syssts_singleoperation = continue`
   - `pd_sts_last_active_side`

4. When both of the current status files have errors:

   Initialize all status files, then execute the method in 1 above (forced startup will be used, regardless of the previous termination mode).

5. When a virtual status file is specified

   Specify the name of the current file.*

* The names of the current files (that were active at the end of the previous operation) can be determined from the following messages:

- `KFPS01001-I`
- `KFPS01010-E`
- `KFPS01011-I`
- `KFPS01063-I`

Of the status files displayed by these messages, the one that is reported in the message that was output most recently is the current file.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for transaction processing are used at the transaction execution site. At the log application site, this operand is ignored, if specified.

54) pd_sts_last_active_side = A | B

Specify this operand if you want to start the back-end server when one of the current files is in an error state. Specify the normal status file for this operand. HiRDB compares the file specified in this operand with the file selected by HiRDB. If the files match, HiRDB copies the contents of the normal status file to secondary File A and File B, then HiRDB switches the spare file in as the current file and starts the unit. If the files do not match, HiRDB (or a unit for a HiRDB/ Parallel Server) is not started.

Conditions

The following operands must be specified:

- `pd_sts_initial_error = continue` or `excontinue`
- `pd_sts_last_active_file`

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for transaction processing are used at the transaction execution site. At the log application site, this operand is ignored, if specified.

55) pd_sts_last_active_subfile = "*logical-file-name*"

~ <identifier> ((1-8 characters))

When Real Time SAN Replication based on the log-only synchronous method is used, this operand specifies the logical file name of the status file for log application processing that is to be used as the current file when HiRDB (or the unit in the case of a HiRDB/Parallel Server) starts. HiRDB compares the file specified in this operand with the current file selected by HiRDB. If the file names match, HiRDB starts; if not, HiRDB does not start. Note that this operand is ignored if HiRDB is started immediately after the database is inherited pursuant to execution of the `pdrisedbto` command.

Conditions

For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*

The following conditions must be satisfied:

- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.
- `continue` or `excontinue` is specified in the `pd_sts_initial_error` operand.
- It cannot be determined if the current file selected by the HiRDB system was the most recent current file during the previous session.

Specification guidelines

1. When starting HiRDB immediately after initializing all status files for log application processing

   Perform the preparations for log application. Once the preparations for log application have been completed successfully, start HiRDB without specifying this operand.

2. When both of the current files are normal

   Specify the name of the current file.# If HiRDB cannot be started even though the name of the current file is specified, the current file may have been initialized. In such a case, first initialize all status files for log application processing, then use the method in 1 above to start HiRDB. Note that in this case, forced startup will be used, regardless of the previous termination mode.

3. When one of the current files has an error

   Use the method in 2 above and specify the following operands:

   - Specify `continue` in the `pd_syssts_singleoperation` operand
   - Specify the `pd_sts_last_active_side_sub` operand

4. When both of the current files have errors

   Perform the preparations for log application. Once the preparations for log application have been completed successfully, start HiRDB without specifying this operand.

5. When a virtual status file for log application processing is specified

   Specify the name of the current file.#

#

The names of the current files (that were active at the end of the previous session) can be determined from the following messages:

- `KFPS01001-I`
- `KFPS01010-E`
- `KFPS01011-I`
- `KFPS01063-I`

Of the status files for log application processing that are displayed in these messages, the one that is reported in the message that was output most recently is the current file.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing are used at the log application site. At the transaction execution site, this operand is ignored, if specified.

56) pd_sts_last_active_side_sub = A | B

When Real Time SAN Replication based on the log-only synchronous method is used, specify this operand if you wish to start HiRDB (or the unit in the case of a HiRDB/Parallel Server) when one of the current files is in error status. In such a case, you specify the normal file in this operand. HiRDB compares the file specified in this operand with the file selected by HiRDB. If their names match, HiRDB copies the contents of the normal file into the secondary file. HiRDB then designates the secondary file as the current file and starts itself. If the file names do not match, HiRDB does not start. This operand is ignored if HiRDB is started immediately after the database was inherited pursuant to execution of the `pdrisedbto` command.

Conditions

For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*

The following conditions must be satisfied:

- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.
- `continue` or `excontinue` is specified in the `pd_sts_initial_error` operand.
- The `pd_sts_last_active_subfile` operand is specified.

Notes

If Real Time SAN Replication based on the log-only synchronous method is used, the server status files for log application processing are used at the log application site. At the transaction execution site, this operand is ignored, if specified.

8.3.17 Operands related to the BES connection holding facility

For details about the BES connection holding facility, see the *HiRDB Version 8 System Operation Guide*.

57) pd_bes_connection_hold = Y | N

This operand is applicable only to a HiRDB/Parallel Server.

Specifies whether to use the BES connection holding facility.

`Y`: The BES connection holding facility is used.

`N`: The BES connection holding facility is not used.

Relationship to the client environment definition

The value of this operand can be changed for each client. To change the operand for a client, specify the `PDBESCONHOLD` operand in the client environment definition. For details about the PDBESCONHOLD operand, see the *HiRDB Version 8 UAP Development Guide*.

Note

When you use the BES connection holding facility, make sure that the following condition is satisfied:

number-of-processes-in-each-back-end-server (*value of the* pd_max_bes_process operand) ≥ *number-of-all-front-end-server-processes* (*value of the* pd_max_users operand × *number-of-front-end-servers*)

If this condition is not satisfied, a shortage in the number of back-end server processes may cause an SQL error. Furthermore, if you plan to execute a program such as a utility while HiRDB is running, allocate the number of back-end server processes required by the utility.

58) pd_bes_conn_hold_trn_interval = *back-end-server-connection-hold-time*

~ <unsigned integer>((0-3600))(seconds)

Specifies the BES connection holding period in seconds.

When the BES connection holding facility is used, HiRDB monitors the period between the termination of a transaction and the execution of the next transaction. If this period is within the specified value, the connection between the front-end server and the back-end server is maintained. However, if this period exceeds the specified value, the connection between the front-end server and the back-end server is terminated after the transaction is terminated.

If 0 is specified for this operand, the period is not monitored. The connection between the front-end server and the back-end server is terminated only when the connection between the front-end server and a client is terminated by SQL DISCONNECT (xa_close if the XA library is being used) or because the value of the PDCWAITTIME operand is exceeded.

8.3.18 Operands related to the work table files

59) pdwork -v "*HiRDB-file-system-area-name*"[,"*HiRDB-file-system-area-name*"]...

~ <pathname of up to 141 characters>

This operand must be specified in order to execute SQL statements.

Specifies the names of HiRDB file system areas for work table files. Work table files are used for temporary storage of information during execution of SQL statements; they are created automatically by HiRDB. For details about the SQL statements that require a work table file, see the *HiRDB Version 8 Description*.

Notes

- Specify in this operand the HiRDB file system area that was initialized using the `pdfmkfs` command.
- If the size of the work table file is large, specify a large HiRDB file system area. For the method of estimating the work table file size, see the *HiRDB Version 8 Installation and Design Guide*.
- The HiRDB file system areas for work table files must be different from the HiRDB file system areas for system files and RDAREAs.
- In the Linux version, you cannot specify a HiRDB file system area created in a character special file.

Operand rules

- At least one HiRDB file system area must be specified.
- A maximum of 16 HiRDB file system areas may be specified.
- This operand may be specified only once in the back-end server definition. If it is specified more than once, the first specification is effective.
- The specified HiRDB file system areas must not be used for other back-end servers.
- When you are specifying multiple HiRDB file system areas, those created in character special files can coexist with those created in regular files.

8.3.19 Operands related to system log file configuration

60) pdlogadfg -d sys -g *file-group-name* [ONL]

Specifies a file group for a system log file. This operand cannot be omitted and must be specified. Use the `pdlogadpf` operand to assign system log files to the file group specified here.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name* ~ <identifier>((1-8 characters))

Specifies a name for the file group. All file group names must be unique within a server.

`ONL`:

Specify this option to enable (open) this file group when HiRDB is running. You can specify 2 to 200 file groups with `ONL` specified.

Operand rule

This operand must be specified at least twice but no more than 200 times.

Notes

Before adding, modifying, or deleting this operand when HiRDB Datareplicator (extracted side) is linked, terminate the corresponding HiRDB Datareplicator. If this operand is modified while HiRDB Datareplicator is running, its extraction process may fail.

61) pdlogadpf -d sys -g *file-group-name* -a "*system-log-file-name*" [-b "*system-log-file-name*"]

Specifies the system log files that comprise the file group. This operand cannot be omitted; it must be specified once for each file group.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name* ~ <identifier>((1-8 characters))

Specifies the file group name specified by the `pdlogadfg` operand. The file group name must be unique within the unit.

-a "*system-log-file-name*" ~ <pathname>((up to 167 characters))

Specifies an absolute path name as the name of the system log file that comprises the file group. Specify the name of the system log file that was initialized using the `pdloginit` command. Note that the system log file name must be unique within the unit.

-b "*system-log-file-name*" ~ <pathname>((up to 167 characters))

Specifies an absolute path name as the name of the system log file B when dual system log files are to be used (`pd_log_dual = Y` specified). If `pd_log_dual = Y` is not specified, the system log file name is invalid, even if it is specified.

Specify the name of the system log file that was initialized using the `pdloginit` command. Note that the system log file name must be unique within the unit.

Notes

Before adding, modifying, or deleting this operand when HiRDB Datareplicator (extracted side) is linked, terminate the corresponding HiRDB Datareplicator. If this operand is modified while HiRDB Datareplicator is running, its extraction process may fail.

8.3.20 Operands related to synchronization point dump file configuration

62) pdlogadfg -d spd -g *file-group-name* [ONL]

Specifies a file group for synchronization point dump files. This operand cannot be omitted and must be specified. Use the `pdlogadpf` operand to assign synchronization point dump files to the file group specified here.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name* ~ <<identifier>((1-8 characters))

Specifies a name for the file group. All file group names within a unit must be server.

`ONL`:

Specify this option to enable (open) this file group when HiRDB is running. You can specify 2 to 30 file groups with `ONL` specified.

Operand rule

This operand must be specified at least twice but no more than 60 times.

63) pdlogadpf -d spd -g *file-group-name* -a "*synchronization-point-dump-file-name*" [-b "*synchronization-point-dump-file-name*"]

Specifies the synchronization point dump files that comprise the file group. This operand cannot be omitted; it must be specified once for each file group.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name*: ~ <identifier>((1-8 characters))

Specifies the file group name specified by the `pdlogadfg` operand. The file group name must be unique within the unit.

-a "*synchronization-point-dump-file-name*": ~ <pathname> ((up to 167 characters))

Specifies an absolute path name as the name of the synchronization point dump file that comprises the file group. Specify the name of the synchronization point dump file that was initialized using the `pdloginit` command. Note that the synchronization point dump file name must be unique within the unit.

-b "*synchronization-point-dump-file-name*": ~ <pathname> ((up to 167 characters))

Specifies an absolute path name as the name of the synchronization point dump file B when dual synchronization point dump files are to be used (`pd_spd_dual = Y` specified). If `pd_spd_dual = Y` is not specified, the synchronization point dump file name is invalid, even if it is specified.

Specify the name of the synchronization point dump file that was initialized using the `pdloginit` command. Note that the synchronization point dump file name must be unique within the unit.

64) pdlogadfg -d ssp -g *file-group-name*[ONL]

When Real Time SAN Replication based on the log-only synchronous method is used, this operand specifies a group of synchronization point dump files for log application processing that is to be used at the log application site. The `pdlogadpf` operand is used to allocate the synchronization point dump files for log application processing.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name*: ~ <identifier> ((1-8 characters))

Specifies the name of a file group. This file group name must correspond to a set of synchronization point dump files for transaction processing.

`ONL`:

Specifies that this file group is to be made available (opened) when HiRDB is running. You can specify 2 to 30 file groups with `ONL` specified.

If Real Time SAN Replication based on the log-only synchronous method is used, when HiRDB is started for the first time at the log application site after the preparations for log application have been performed, the file groups that need to be inherited from the transaction execution site become available when the single server starts, regardless of the specification of `ONL`.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.

Operand rules

You can specify 2 to 60 instances of this operand.

Notes

Specify a `pdlogadfg -d ssp` operand for each file group specified in a `pdlogadfg -d spd` operand.

65) pdlogadpf -d ssp -g *file-group-name* -a "*synchronization-point-dump-file-name-for-log-application-processing*" [-b "*synchronization-point-dump-file-name-for-log-application-processing*"]

Specifies synchronization point dump files for log application processing that constitute a file group. You must specify one `pdlogadpf -d ssp` operand for each file group.

Make sure that the `pdlogadfg` and `pdlogadpf` operands are specified in this order; otherwise, an error results.

-g *file-group-name*: ~ <identifier> ((1-8 characters))

Specifies the name of a file group specified in a `pdlogadfg` operand. This file group name must correspond to a set of primary synchronization point dump files for transaction processing.

-a "*synchronization-point-dump-file-name-for-log-application-processing*": ~ <path name> ((up to 167 characters))

Specifies the absolute path name of a synchronization point dump file for log application processing that is to be included in the file group. This must be the name of a synchronization point dump file for log application processing that was created during the preparations for log application. Note that each synchronization point dump file name for log application processing must be unique within the same unit.

-b "*synchronization-point-dump-file-name-for-log-application-processing*": ~ <path name> ((up to 167 characters))

When dual synchronization point dump files for log application processing are used (`Y` is specified in the `pd_spd_dual` operand), this option specifies the absolute path name of a secondary synchronization point dump file for log application processing. If `Y` is not specified in the `pd_spd_dual` operand, the `-b` option is ignored, if specified. You must specify in this option the name of a synchronization point dump file for log application processing created during the preparations for log application. Note that each synchronization point dump file for log application processing must be unique within the same unit.

Conditions

- For details about the required platform, see the *HiRDB Version 8 Disaster Recovery System Configuration and Operation Guide.*
- `Y` must be specified in the `pd_rise_use` operand, and also `syssync` must be specified in the `pd_rise_pairvolume_combination` operand.

Notes

Specify a `pdlogadpf -d ssp` operand for each file group specified in a `pdlogadfg -d ssp` operand.

8.3.21 Operands related to Plug-ins

66) pdplgprm -n *plug-in-name* [-s *shared-memory-size*]

Specifies the name of a plug-in and the size of the memory to be shared by the plug-in. This operand should be omitted if no plug-ins are to be used or no plug-ins will run in back-end server.

Conditions

The plug-in specified here must have been registered in HiRDB with the `pdplgrgst` command.

-n *plug-in-name*: ～ <identifier>((1-30 characters))

For details about the names of plug-ins that can be specified here, see the manuals for the plug-ins.

-s *shared-memory-size*: ～ <unsigned integer> ((1-2000000)) <<0>> (KB)

Specifies in kilobytes the size of the shared memory to be used by the plug-in. For details about the size of the shared memory to be used by the plug-in, see the manual for the applicable plug-in.

8.3.22 Operands related to HiRDB External Data Access facility (environment variables)

For details on the HiRDB External Data Access facility, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

Note that some platforms on which HiRDB is run do not support the HiRDB External Data Access facility. For details, see *HiRDB External Data Access* in the manual *HiRDB Version 8 Description*.

67) putenv *environment-variables-needed-by-foreign-server*

This operand specifies the environment variables required by a foreign server. For the required environment variables, see the documentation for the DMBS of the foreign server to be connected.

If the required environment variables are not specified, an error may occur during access to the foreign server.

Condition

HiRDB External Data Access is required.

Chapter

9. UAP Environment Definition

This chapter explains the operands of the UAP environment definition.

This chapter contains the following sections:

9.1 Operand formats

A UAP environment definition defines the execution environment of a UAP. This section explains the formats used to specify the operands of a UAP environment definition. Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *9.2 Operand explanations*.

No.	Format	Modifiable after forced or abnormal termination?	Modifiable after planned termination ?
1	[set pd_uap_wait = Y | N]	Y	Y
2	[{{pdlbuffer -a *local-buffer-name* {-r *RDAREA-name*[,*RDAREA-name*]... | -i *authorization-identifier*.*index-identifier*} -n *buffer-sector-count* [-p *batch-input-maximum-page-count*]}}]	Y	Y

Y: Yes, specification value can be modified.

9.2 Operand explanations

1) pd_uap_wait = Y | N

Specifies the action to be taken by the UAP when the RDAREA or index to be accessed using a local buffer is being used by another user.

`Y`: Waits for the other user to terminate processing. The UAP goes into a wait state.

`N`: If the RDAREA or index to be accessed using a local buffer is being used by another user, an error is returned to the UAP.

If this operand is specified together with two or more `pdlbuffer` operands, the specification of this operand is valid for all the local buffers specified.

Condition

The `pdlbuffer` operand must be specified.

2) `pdlbuffer -a` *local-buffer-name*

{`-r` *RDAREA-name*[,*RDAREA-name*]...| `-i` *authorization-identifier.index-identifier*}

`-n` *buffer-sector-count*

[`-p` *batch-input-maximum-page-count*]

Specifies the local buffer used by a UAP. A local buffer is an area used for inputting and outputting table or index data located in a disk, and is allocated in the process private memory of a single server or back-end server. The maximum number of local buffers you can define is 100. For a HiRDB/Parallel Server, you can define up to 100 local buffers for each back-end server.

When a local buffer is specified, various types of resources are locked according to the specification of the `pdlbuffer` operand as shown as follows.

pdlbuffer operand specification	Locked resource	Lock mode
The `-r` option of the `pdlbuffer` operand is specified.	Specified RDAREA	`EX`
The `-i` option of the `pdlbuffer` operand is specified.	RDAREA for index	`SU`
	Table for index	`EX`
	Specified index	`EX`

Condition

A global buffer must first be allocated to the RDAREA or index to be specified as a local buffer, using the `pdbuffer` operand.

Application criterion

Define a local buffer when all of the following conditions are satisfied:

- A large volume of data is searched or updated.
- The access target RDAREA is not accessed by other UAPs.

Because UAPs that are always connected to HiRDB have a major impact on the system (memory shortage or process lock), do not define local buffers for them.

Note

Note the following for a HiRDB/Parallel Server:

- If a server process is abnormally terminated when a local buffer is being used, the unit may be abnormally terminated, and the abort code `Phb3008` may be output.
- If a page is being updated when the server process is abnormally terminated, recovery using a rollback process may not be possible. In this case, recovery is performed when the unit is restarted.
- For the processing by HiRDB and the action to be taken when an error occurs when a local buffer is being used, see the *HiRDB Version 8 System Operation Guide*.

`-a` ***local-buffer-name***

~ ***<identifier>*****((1-16 characters))**

Specifies a local buffer name. The name specified here must be unique. However, the same name may be specified for local buffers that are in different UAP environment definition files.

`-r` ***RDAREA-name*****[,*****RDAREA-name*****]...**

~ ***<identifier>*****((1-30 characters))**

Specifies the name of the RDAREA to which a local buffer for data is allocated. Only user RDAREA names may be specified.

The name specified here must be unique. However, the same name may be specified for RDAREAs that are in different UAP environment definition files.

Specification guidelines

When multiple RDAREAs are allocated to a single local buffer, the maximum page size of the RDAREAs becomes the buffer size. Therefore, allocating RDAREAs having the same or approximately the same size to the local buffer can reduce the number of inputs and outputs. However, with the types of RDAREAs described as follows, allocating them to different local buffers can reduce the number of inputs and outputs even if these RDAREAs have the same size.

- RDAREAs that store tables for different purposes
- RDAREAs that are accessed randomly or sequentially most of the time

Operand rules

- If the RDAREA name includes a character that is not alphanumeric, enclose it in quotation marks.
- The maximum number of RDAREAs you can define for a single local buffer is 3,200.

`-i` ***authorization-identifier.index-identifier***

***authorization-identifier*: ~ <*identifier*>((1-8 characters))**

***index-identifier*: ~ <*identifier*>((1-30 characters))**

Specifies the index name (*authorization-identifier.index-identifier*) to which the local buffer for the index is allocated.

The name specified here must be unique. However, the same name may be specified for indexes that are in different UAP environment definition files.

Operand rule

If the authorization identifier or index identifier includes a character that is not alphanumeric, enclose it in quotation marks.

Specification guidelines

Specify an index that is frequently used. Allocating a local buffer to a frequently used index increases the degree of memory residency for index pages, and thus can reduce the number of inputs and outputs.

In particular, allocating an index defined in a cluster key or unique key to a local buffer can have a major positive effect. Note that because the index identifier of a cluster key is determined by HiRDB, check the index identifier by searching the dictionary table (`INDEX_NAME` column of the `SQL_INDEXES` table) after a table has been defined.

`-n` ***buffer-sector-count***

~ <unsigned integer>((4-125000))

Specifies the number of local buffer sectors.

Specification guidelines

- A local buffer is allocated in the process private memory. Therefore, allocating an unnecessarily large area to the local buffer causes frequent paging when the rest of the memory is used, resulting in poor performance.
- Note that specifying too large a buffer sector count may make it impossible to allocate process private memory.
- For a sequentially-accessing UAP, no buffering effect can be expected. Therefore, specify the minimum value of 4 for the buffer sector count.

`-p` ***batch-input-maximum-page-count***

~ **<unsigned integer>((2-256))**

Specifies the maximum number of pages that can be input by the prefetch facility in a single batch. If this operand is omitted, the prefetch facility does not work.

Specification guidelines

By taking the cost-performance tradeoff between memory size and input/out time reduction effect into consideration, specify a value that satisfies the following condition:

$a \times b$ = 64-128 (KB)

a: Size of the RDAREA that stores the data or index to be prefetched

b: Maximum number of pages that can be input in a single batch

If the segment size of the RDAREA is 1, specifying this option has no effect. Therefore, do not specify it.

Chapter

10. Foreign Server Information Definition (When a Foreign Server Is HiRDB)

This chapter explains the foreign server information definition when a foreign server is HiRDB. Create this definition for using the HiRDB External Data Access facility.

10.1 Operand formats

The foreign server information definition defines the execution environment for the HiRDB External Data Access facility. Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *10.2 Operand explanations*.

- To users who are creating a foreign server information definition for the first time

  First, determine the values to be specified for the operands shown in bold. Basically, once you specify these operands, you can use the HiRDB External Data Access facility.

- Operands you can modify before restarting HiRDB

  You can modify all operands in the foreign server information definition before restarting HiRDB (following planned termination, forced termination, or abnormal termination).

No.	Format	Operand category
1	`[set pd_hb_db_con = STARTUP	ACCESS]`	Connection to and disconnection from a foreign server
2	`[set pd_hb_db_dis_con = SHUTDOWN	COMMIT]`	
3	`[set pd_hb_e_mode = ALL	NONE]`	Foreign server error information
4	`[set pd_hb_e_code = SQLCODE[,SQLCODE]...]`	
5	`[set pd_hb_e_size =` *foreign-server-error-information-file-size*`]`	

No.	Format	Operand category
6	[set pd_hb_l_path = *foreign-server-interface-trace-information-file-output-destination-directory*]	Foreign server interface trace information
7	[set pd_hb_l_size = *foreign-server-interface-trace-information-file-size*]	
8	[set pd_hb_l_prm = Y | N]	
9	[set pd_hb_l_prm_size = *maximum-data-size-of-?-parameter-information-to-be-output-to foreign-server-interface-trace-information-file*]	
10	[set pd_hb_l_mode = ALL | PARTIAL]	
11	[set pd_hb_ary_fec_num = *number of rows-to-be-fetched-at-once-with-FETCH-facility-that-uses-array*]	FETCH
12	[putenv PDCURSORLVL 0 | 1 | 2]	

No.	Format	Operand category
13	putenv PDHOST *foreign-HiRDB-host-name*[,*foreign-HiRDB-secondary-system-host-name*]	Connection to a foreign server
14	[putenv PDFESHOST *foreign-HiRDB-front-end-server-host-name*[:*foreign-HiRDB-front-end-server-port-number*][,*foreign-HiRDB-secondary-system-front-end-server-host-name*[:*foreign-HiRDB-secondary-system-front-end-server-host-name*]]]	
15	putenv PDNAMEPORT *foreign-HiRDB-port-number*	
16	[putenv PDSERVICEPORT *foreign-HiRDB-high-speed-connection-port-number*[,*foreign-HiRDB-secondary-system-high-speed-connection-port-number*]]	
17	[putenv PDSERVICEGRP *foreign-HiRDB-server-name*]	
18	[putenv PDSRVTYPE WS | PC]	
19	[putenv PDCLTRCVPORT *receive-port-range-to-be-reserved*]	
20	[putenv PDCLTRCVADDR *back-end-server-IP-address-or-host-name-for-connecting-to-foreign-server*]	
21	[putenv PDAUTORECONNECT YES | NO]	
22	[putenv PDRCCOUNT *maximum-number-of-CONNECT-retries-using-automatic-reconnect-facility*]	
23	[putenv PDRCINTERVAL *CONNECT-retry-interval-for-using-automatic-reconnect-facility*]	
24	[putenv PDDBLOG ALL | NO]	Database update log
25	[putenv PDCLTGRP *client-group-name*]	Client group

No.	Format	Operand category
26	`[putenv PDIPC MEMORY | DEFAULT]`	Communication processing
27	`[putenv PDSENDMEMSIZE` *memory-size-for-sending-data*`]`	
28	`[putenv PDRECVMEMSIZE` *memory-size-for-receiving-data*`]`	
29	`[putenv PDTCPCONOPT 0 | 1]`	
30	`[putenv PDKALVL 0 | 1 | 2]`	
31	`[putenv PDKATIME` *interval-for-sending-packets-to-foreign-HiRDB*`]`	
32	`[putenv PDNODELAYACK YES | NO]`	
33	`[putenv PDCWAITTIME` `[`*maximum-wait-time-1-for-HiRDB-client-used-by-HiRDB-External-Data-Access-Adapter*`][,`*maximum-wait-time-2-for-HiRDB-client-used-by-HiRDB-External-Data-Access-Adapter*`]]`	System monitoring
34	`[putenv PDSWAITTIME` `[`*maximum-wait-time-1-for-foreign-HiRDB*`][,`*maximum-wait-time-2-for-foreign-HiRDB*`]]`	

No.	Format	Operand category
35	[putenv PDCLTPATH *SQL-trace-file-and-error-log-file-output-destination-directory*]	Troubleshooting information
36	[putenv PDSQLTRACE *SQL-trace-file-size*]	
37	[putenv PDSQLTEXTSIZE *size-of-SQL-statements-to-be-output-to-SQL-trace-file*]	
38	[putenv PDSQLEXECTIME YES | NO]	
39	[putenv PDUAPERLOG *error-log-file-size*]	
40	[putenv PDPRMTRC YES | NO]	
41	[putenv PDPRMTRCSIZE *maximum-data-size-of-?-parameter-information-inside-SQL-trace*]	
42	[putenv PDTRCMODE ERR | NONE]	
43	[putenv PDTRCPATH *dynamic-SQL-trace-file-output-destination-directory*]	
44	[putenv PDUAPREPLVL [s][u][p][r] | a]	
45	[putenv PDREPPATH *UAP-statistical-report-output-destination-directory*]	
46	[putenv PDSQLTRCOPENMODE CNCT | SQL]	
47	[putenv PDVWOPTMODE *access-path-information-acquisition-mode*]	
48	[putenv PDLOCKLIMIT *upper-limit-for-lock-requests-to-foreign-HiRDB*]	Lock
49	[putenv PDDBACCS *RDAREA-generation-number-of-foreign-HiRDB*]	Inner replica facility
50	[putenv PDDBORGUAP YES | NO]	

No.	Format	Operand category
51	[putenv PDSQLOPTLVL *SQL-optimization-option-for-foreign-HiRDB*[,*SQL-optimization-option-for-foreign-HiRDB*]...]	SQL optimization
52	[putenv PDADDITIONALOPTLVL *SQL-extension-optimization-option-for-foreign-HiRDB*[,*SQL-extension-optimization-option-for-foreign-HiRDB*]...]	
53	[putenv PDAGGR *number-of-groups-generated-for-foreign-HiRDB-grouping*]	
54	[putenv PDHASHTBLSIZE *hash-table-size-for-hash-join-subquery-hash-execution-by-foreign-HiRDB*]	
55	[putenv PDUAPEXERLOGUSE YES | NO]	Facility for output of extended SQL error information
56	[putenv PDUAPEXERLOGPRMSZ *maximum-data-size-of-parameter-information-to-be-output-to-error-log-file-and-SQL-error-report-file*]	
57	[putenv PDBESCONHOLD YES | NO]	BES connection holding facility
58	[putenv PDBESCONHTI *back-end-server-connection-hold-time*]	
59	[putenv PDHATRNQUEUING NO]	System switchover facility

10.2 Operand explanations

10.2.1 Operands related to connection to and disconnection from a foreign server

1) pd_hb_db_con = STARTUP | <u>ACCESS</u>

Specifies the timing to connect to a foreign server.

`STARTUP`:

Connects to a foreign server when a back-end server process is started.

When `STARTUP` is specified, the foreign server is connected even when there is no processing involving it. Therefore, if there is a limit on the number of foreign servers that can be connected, specify `ACCESS`.

`ACCESS`:

Connects to a foreign server only when a back-end server process requests access to the foreign server.

2) pd_hb_db_dis_con = SHUTDOWN | <u>COMMIT</u>

Specifies the timing to disconnect from a foreign server.

`SHUTDOWN`:

Disconnects from the foreign server when a back-end server process is terminated.

When `SHUTDOWN` is specified, the foreign server is connected even when there is no processing involving it. Therefore, if there is a limit on the number of foreign servers that can be connected, specify `COMMIT`.

`COMMIT`:

Disconnects from the foreign server immediately after `COMMIT` is issued to the foreign server.

10.2.2 Operands related to foreign server error information

3) pd_hb_e_mode = <u>ALL</u> | NONE

Specifies whether to output foreign server error information. For details on foreign server error information, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

`ALL`: Outputs foreign server error information.

`NONE`: Does not output foreign server error information.

4) pd_hb_e_code = SQLCODE [,SQLCODE]...

~ <signed integer>((-214748364 to -1))

For this operand, specify the foreign server's SQLCODE (value returned to SQLCODE of the SQL Communications Area SQLCA). The error message corresponding to the SQLCODE specified here is not output to the foreign server error information.

If this operand is omitted, all error messages corresponding to all SQLCODEs are output to the foreign server error information file. For details on SQLCODEs, see the documentation for each DBMS.

Operand rules

- When specifying multiple SQLCODEs, delimit them using commas.
- The maximum number of SQLCODEs that can be specified is 10.

5) pd_hb_e_size = ***foreign-server-error-information-file-size***

~ <unsigned integer>((0, 4-4096)) <<32>> (KB)

Specifies the size of the foreign server error information file (units: KB). To not limit the size, specify `0`.

10.2.3 Operands related to foreign server interface trace information

6) pd_hb_l_path = ***foreign-server-interface-trace-information-file-output-destination-directory***

~ <pathname> <<$PDDIR/spool/>>

Specifies the output destination directory for foreign server interface trace information. Foreign server interface trace information is output to this directory. Two output files are created under this directory. Their names are `pdhl`*foreign-server-name-1* and `pdhl`*foreign-server-name-2*.

For details on foreign server interface trace information, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

7) pd_hb_l_size = ***foreign-server-interface-trace-information-file-size***

~ <unsigned integer>((0, 4-4096)) <<1024>> (KB)

Specifies the size of the foreign server interface trace information file (units: KB). To not limit the size, specify `0`.

If the file size exceeds the value specified for this operand, the files are swapped and trace information is output to the other file.

Specification guideline

On average, 300 bytes of information is output per trace (excluding the ?

parameter information).

Notes

- For a UAP executed by specifying the `PDSQLTRACE` operand of the client environment definition, foreign server interface trace information is collected for each foreign server. Because multiple transactions write data into a single file, a lock is applied during a writing operation. Therefore, collecting foreign server interface trace information lowers both transaction processing performance and concurrent executability.
- Depending on the access status of the foreign server, the file size may exceed the value specified for this operand. Therefore, collect foreign server interface trace information onto a disk with ample free space.

8) pd_hb_l_prm = Y | N

Specifies whether to output ? parameter information in foreign server interface trace information.

`Y`: Outputs ? parameter information.

`N`: Does not output ? parameter information.

9) pd_hb_l_prm_size = *maximum-data-size-of-?-parameter-information-to-be-output-to foreign-server-interface-trace-information-file*

~ <unsigned integer>((4-4096)) <<256>> (Bytes)

Specifies the maximum data size (units: bytes) of the ? parameter information to be output in the foreign server interface trace information file.

Condition

`pd_hb_l_prm = Y` must be specified.

10) pd_hb_l_mode = ALL | PARTIAL

Specifies whether to output all foreign server interface trace information.

`ALL`: Outputs all information.

`PARTIAL`: Outputs the information on only the facilities that communicate with the foreign server.

10.2.4 Operands related to FETCH

11) pd_hb_ary_fec_num = *number of rows-to-be-fetched-at-once-with-FETCH-facility-that-uses-array*

~ <unsigned integer>((1-30000))

Specifies the number of rows to be fetched at one time with FETCH that uses an

array.

If the number of search result rows from a foreign table can be predicted to some degree, specify this operand. The communication overhead between HiRDB and the foreign server can be expected to decrease. For the SQL execution time and the number of FETCH operations to a foreign table based on the array FETCH count to the foreign server, check the statistical information related to foreign server usage. For details on the statistical information related to foreign server usage, see the *Statistics analysis utility* in the manual *HiRDB Version 8 Command Reference*.

If this operand is omitted, HiRDB determines automatically a value that is optimized for the HiRDB processing.

Specification guidelines

- If the specification value is too small, the number of communications between HiRDB and the foreign server increases, resulting in a greater overhead.
- If the specification value is too large, too large a memory area is used, resulting in a memory resource shortage.
- If the number of search result rows from a foreign table varies depending on the job, Hitachi recommends that this operand be omitted. Whether omitting this operand is advantageous depends on whether one or more rows of data collected from the foreign table fit in the communication buffer size specified by the `pd_sql_send_buff_size` operand. If the data fit, HiRDB efficiently processes the data collected from the foreign table. If the data does not fit, HiRDB suppresses the memory used for accessing the foreign table.

12) PDCURSORLVL <u>0</u> | 1 | 2

Specifies the timing for issuing a cursor open/close request to the HiRDB server when the cursor is used for search processing.

The function of and specification method for this operand are the same as for the `PDCURSORLVL` operand in the client environment definition. For details about the `PDCURSORLVL` operand, see the *HiRDB Version 8 UAP Development Guide*. For an external server with a different HiRDB version, see the applicable manual.

`0`:

When the HiRDB client receives a cursor open/close request from an application, it issues the execution request as is to the HiRDB server.

`1`:

If there is no search data, the HiRDB server returns `SQLCODE=100` and at the same time closes the cursor without a request from the HiRDB client. When the HiRDB client receives a cursor close request from an application and the HiRDB client has already detected `SQLCODE=100`, it does not issue a cursor close request to the HiRDB server; the HiRDB client issues a cursor close request only if it has not detected `SQLCODE=100`.

For a cursor open request, the processing is the same as when `0` is specified.

`2`:

When the HiRDB client receives a cursor open request from an application, it does not issue the execution request to the HiRDB server; instead, the HiRDB client issues a cursor open request when the first fetch request is issued.

For a cursor close request, the processing is the same as when `1` is specified.

10.2.5 Operands related to connection to a foreign server

Specifying the operands explained here can shorten the time to connect to a foreign HiRDB server. The following table shows a list of the operands necessary for connecting to a foreign HiRDB server.

Operand name	When the foreign server is a HiRDB/ Single Server		When the foreign server is a HiRDB/Parallel Server				
	Regular connection	High-speed connection	Single front-end server		Multi front-end servers		
			Regular connection	High-speed connection	Regular connection	Connection to a specific front-end server	
						FES host direct	High-speed
PDHOST	Y	Y	Y	Y	Y	Y	Y
PDFESHOST	N	N	N	Y	N	Y	Y
PDNAMEPORT	Y	Y	Y	Y	Y	Y	Y
PDSERVICEPORT	N	Y	N	Y	N	N	Y
PDSERVICEGRP	N	Y	N	Y	N	Y	Y

Legend:

Y: Must be specified.

N: Need not be specified.

13) PDHOST *foreign-HiRDB-host-name*[,*foreign-HiRDB-secondary-system-host-name*]

~ **<host name>**

Specifies the host name of a foreign HiRDB (HiRDB at a foreign server).

If the foreign HiRDB is a HiRDB/Single Server, this operand specifies the host name of the server machine where the single server is located. If the foreign HiRDB is a HiRDB/Parallel Server, this operand specifies the host name of the server machine where the system manager is located.

The function and specification method of this operand are the same as those of the `PDHOST` operand of the client environment definition. For details on the `PDHOST` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

14) PDFESHOST *foreign-HiRDB-front-end-server-host-name*[:*foreign-HiRDB-front-end-server-port-number*][,*foreign-HiRDB-secondary-system-front-end-server-host-name*[:*foreign-HiRDB-secondary-system-front-end-server-port-number*]]

~ **<identifier>**

Specify this operand if the foreign HiRDB is a multi front-end server. Specify the host name of the connection target front-end server for the foreign HiRDB. If this operand is omitted, the foreign HiRDB decides to which front-end server to be connected.

The function and specification method of this operand are the same as those of the `PDFEHOST` operand of the client environment definition. For details on the `PDFEHOST` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

15) PDNAMEPORT *foreign-HiRDB-port-number*

~ **<unsigned integer>((5001-65535))**

Specifies the port number of the foreign HiRDB (the value of the `pd_name_port` operand of the foreign HiRDB). This operand is required and cannot be omitted.

The function and specification method of this operand are the same as those of the `PDNAMEPORT` operand of the client environment definition. For details on the `PDNAMEPORT` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

16) PDSERVICEPORT *foreign-HiRDB-high-speed-connection-port-number*[,*foreign-HiRDB-secondary-s*

***ystem-high-speed-connection-port-number*]**

~ <unsigned integer>((5001-65535))

Specifies the port number for high-speed connection. Specify a value for the `pd_service_port` operand of the foreign HiRDB. Specifying this operand can shorten the time needed to connect to the foreign HiRDB.

The function and specification method of this operand are the same as those of the `PDSERVICEPORT` operand of the client environment definition. For details on the `PDSERVICEPORT` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

17) PDSERVICEGRP *foreign-HiRDB-server-name*

~ <character string>

Specifies the server name of the foreign HiRDB. If the foreign HiRDB is a HiRDB/Single Server, this operand specifies a single server name. If the foreign HiRDB is a HiRDB/Parallel Server, this operand specifies a front-end server name. If the foreign HiRDB is a multi front-end server, this operand specifies the name of the connection target front-end server.

The function and specification method of this operand are the same as those of the `PDSERVICEGRP` operand of the client environment definition. For details on the `PDSERVICEGRP` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

18) PDSRVTYPE <u>WS</u> | PC

Specifies a foreign HiRDB type.

`WS`: Specify this option when the foreign HiRDB is the UNIX version.

`PC`: Specify this option when the foreign HiRDB is the Windows version.

19) PDCLTRCVPORT *receive-port-range-to-be-reserved*

~ <unsigned integer>((5001-65535))

Specifies the reservation range of the port numbers to be used for communication with the foreign HiRDB. If this operand is omitted, the OS automatically assigns available ports. Therefore, normally you need not specify this operand.

Operand specification methods

Specification examples of this operand follow.

- To use a single port number (10000)

```
PDCLTRCVPORT 10000
```

- To specify a port number range (10000-105000)

```
PDCLTRCVPORT 10000 - 10500
```

Specification guideline

Specify this operand so that there will be no port number shortage. The number of port numbers necessary is as follows:

Number of port numbers necessary > Value of the `pd_max_bes_process` operand of the foreign HiRDB

Note

If a unit contains multiple back-end servers for connecting to the foreign server, be careful about the value to be specified for this operand. If the value specified for this operand is duplicated in foreign server definitions (that is, if the same range of port numbers is reserved), the overlapping range of port numbers is used by the back-end servers. Consequently, a port number shortage may occur.

The function and specification method of this operand are the same as those of the `PDCLTRCVPORT` operand of the client environment definition. For details on the `PDCLTRCVPORT` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

20) PDCLTRCVADDR *back-end-server-IP-address-or-host-name-for-connecting-to-foreign-server*

~ **<unsigned integer>** or **<*identifier*>**

Specify this operand if multiple communication routes are provided in the server machine (where the back-end server for connecting to the foreign server is located) is used. Specify the IP address or host name of the communication route to be used for communication with the foreign server.

If this operand is omitted, the IP address of the standard host name of the server machine where the back-end server for connecting to the foreign server is located is used.

The function and specification method of this operand are the same as those of the `PDCLTRCVADDR` operand of the client environment definition. For details on the `PDCLTRCVADDR` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

21) PDAUTORECONNECT YES | <u>NO</u>

Specifies whether to use the automatic reconnect facility for the foreign HiRDB. For details on the automatic reconnect facility, see the *HiRDB Version 8 UAP*

Development Guide.

YES: Uses the automatic reconnect facility.

NO: Does not use the automatic reconnect facility.

The CONNECT retry count and CONNECT retry interval for reconnection are specified by the PDRCCOUNT and PDRCINTERVAL operands of the foreign server information definition.

The function and specification method of this operand are the same as those of the PDAUTORECONNECT operand of the client environment definition. For details on the PDAUTORECONNECT operand, see the *HiRDB Version 8 UAP Development Guide*.

22) PDRCCOUNT
maximum-number-of-CONNECT-retries-using-automatic-reconnect-facility

~ <unsigned integer>((1-200)) <<5>>

Specifies the maximum number of CONNECT retries for reconnecting using the automatic reconnect facility. If the number of CONNECT retries exceeds the value of this operand, the attempt to connect to the foreign server is cancelled.

Condition

The PDAUTORECONNECT operand of the foreign server information definition must be set to YES.

23) PDRCINTERVAL
CONNECT-retry-interval-for-using-automatic-reconnect-facility

~ <unsigned integer>((0-600)) <<5>> (seconds)

Specifies the CONNECT retry interval (units: seconds) to be used for reconnecting using the automatic reconnect facility.

Condition

The PDAUTORECONNECT operand of the foreign server information definition must be set to YES.

10.2.6 Operands related to the database update log

24) PDDBLOG <u>ALL</u> | NO

Specifies whether to collect the update log of the database of the foreign server.

ALL:

Executes an SQL statement for the foreign server in the log acquisition mode (collects the update log of the database).

NO:

Executes an SQL statement for the foreign server in the no-log mode (does not collect the update log of the database).

10.2.7 Operands related to client group

25) PDCLTGRP *client-group-name*

~ <character string> ((1 character))

Specify this operand if the foreign HiRDB uses the connection frame guarantee facility for a client group. Specify a client group name for this operand. Use a single capital letter to specify the client group name specified for the `pdcltgrp` operand of the foreign HiRDB. If a lower-case letter is specified, it is assumed to be an upper-case letter.

If the `pdcltgrp` operand of the foreign HiRDB is not specified, or if a client group name not specified in the `pdcltgrp` operand is specified, the specification of the `PDCLTGRP` operand is invalid.

The function and specification method of this operand are the same as those of the `PDCLTGRP` operand of the client environment definition. For details on the `PDCLTGRP` operand, see the *HiRDB Version 8 UAP Development Guide*. For details on the connection frame guarantee facility for a client group, see the *HiRDB Version 8 System Operation Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

10.2.8 Operands related to communication processing

26) PDIPC MEMORY | DEFAULT

Specifies the inter-process communication method to be used when the back-end server for connecting to the foreign server and the foreign HiRDB (a single server or the unit containing a front-end server) are located on the same server machine.

`MEMORY`:

Uses memory for the inter-process communication. This is called the inter-process memory communication facility. This facility increases the speed of communication with the foreign HiRDB.

`DEFAULT`:

Uses the default communication method (TCP/IP or PIPE) of each OS for inter-process communication.

The function and specification method of this operand are the same as those of the `PDIPC` operand of the client environment definition. For details on the `PDIPC` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

27) PDSENDMEMSIZE *memory-size-for-sending-data*

~ <unsigned integer>((4-2097152)) <<16>> (KB)

Specify this operand if you use the inter-process memory communication facility (`MEMORY` is specified for the `PDIPC` operand).

Specify the size of the data storage area (units: KB) to be used for sending data to the foreign server. Specify a multiple of 4.

The function and specification method of this operand are the same as those of the `PDSENDMEMSIZE` operand of the client environment definition. For details on the `PDSENDMEMSIZE` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

28) PDRECVMEMSIZE *memory-size-for-receiving-data*

~ <unsigned integer>((4-2097152)) <<32>> (KB)

Specify this operand if you use the inter-process memory communication facility (`MEMORY` is specified for the `PDIPC` operand).

Specify the size of the data storage area (units: KB) to be used for receiving data from the foreign server. Specify a multiple of 4.

The function and specification method of this operand are the same as those of the `PDRECVMEMSIZE` operand of the client environment definition. For details on the `PDRECVMEMSIZE` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

29) PDTCPCONOPT <u>0</u> | 1

Specifies whether the number of TCP ports to be used by HiRDB External Data Access Adapter for communication between a HiRDB client and a foreign HiRDB (Version 06-02 or newer) be reduced. Depending on the TCP protocol specifications, TCP ports may not be usable for new TCP connection (by going into the TIME_WAIT state) for a certain amount of time (1-4 minutes) after TCP connection is terminated. Ports in the TIME_WAIT state are used by the TCP connection that has been terminated.

`0`: Does not reduce the number of TCP ports in the TIME_WAIT state.

`1`: Reduces the number of TCP ports in the TIME_WAIT state.

The function and specification method of this operand are the same as those of the `PDTCPCONOPT` operand of the client environment definition. For details on the `PDTCPCONOPT` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

30) PDKALVL 0 | 1 | 2

Specifies whether to send packets from the HiRDB client used by HiRDB External Data Access Adapter to a foreign HiRDB.

`0`: Does not send packets to the foreign HiRDB.

`1`: Regularly sends packets to the connection route with the foreign HiRDB.

`2`: Regularly sends packets to the connection route with the foreign HiRDB and receives return packets from the foreign HiRDB.

If `1` or `2` is specified for this operand, a packet send thread is generated for each connection with the foreign HiRDB. The packet send interval is specified by the `PDKATIME` operand of the foreign server information definition.

Specification guidelines

- Some network management applications, such as routers and firewalls, terminate connection if packets are not received within a specified amount of time. Consequently, web applications that wait for service requests while being connected to a foreign HiRDB are disconnected from the foreign HiRDB when the wait state exceeds the predetermined time limit. To avoid this kind of problem, specify `1` or `2`.
- When maximum wait time is not specified for a foreign HiRDB (the first argument of the `PDSWAITTIME` operand of the foreign server information definition is omitted and `0` is specified for the second argument), a server process may remain in the foreign HiRDB, if the server machine on which the HiRDB client used by HiRDB External Data Access Adapter is located is abnormally terminated, or if a network error occurs. Specifying `2` for this operand can prevent these server processes from remaining.

The function and specification method of this operand are the same as those of the `PDKALVL` operand of the client environment definition. For details on the `PDKALVL` operand, see the *HiRDB Version 87 UAP Development Guide*.

31) PDKATIME *interval-for-sending-packets-to-foreign-HiRDB*

~ <unsigned integer>((60-65535)) <<3000>> (seconds)

Specifies the interval (units: seconds) at which to send packets from the HiRDB client used by HiRDB External Data Access Adapter to a foreign HiRDB.

Condition

The `PDKALVL` operand of the foreign server information definition must be set to `1` or `2`.

The function and specification method of this operand are the same as those of the `PDKATIME` operand of the client environment definition. For details on the

PDKATIME operand, see the *HiRDB Version 8 UAP Development Guide*.

32) PDNODELAYACK YES | NO

This operand is specified when an external HiRDB being used is the AIX 5L version.

The operand specifies whether or not immediate acknowledgment is to be used when data is received during communication between the HiRDB server machine and the HiRDB client machine.

The function of and specification method for this operand are the same as for the PDNODELAYACK operand in the client environment definition. For details about the PDNODELAYACK operand, see the *HiRDB Version 8 UAP Development Guide*. For an external server with a different HiRDB version, see the applicable manual.

YES: Use immediate acknowledgment.

NO: Do not use immediate acknowledgment.

10.2.9 Operands related to system monitoring

33) PDCWAITTIME [*maximum-wait-time-1-for-HiRDB-client-used-by-HiRDB-External-Data-Access-Adapter*][,*maximum-wait-time-2-for-HiRDB-client-used-by-HiRDB-External-Data-Access-Adapter*]

The function of this operand is the same as that of the PDCWAITTIME operand of the client environment definition. For details on the PDCWAITTIME operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

maximum-wait-time-1-for-HiRDB-client-used-by-HiRDB-External-Data-Access-Adapter:

~ <signed integer>((-65534 to 0))<<0>> (seconds)

Specifies the number of seconds to be deducted from the value of the PDCWAITTIME operand of the client environment definition. This operand determines the maximum amount of time the HiRDB client used by HiRDB External Data Access Adapter waits for a foreign HiRDB to respond following a processing request.

Operand specification methods

The maximum wait time is determined as follows:

Maximum wait time = *value-of-this-operand* + *value-of-PDCWAITTIME-operand-of-client-environment-definition*

For example, if -10 and 100 are specified for this operand and the PDCWAITTIME operand of the client environment definition, respectively, the maximum wait time is 90 seconds (100 - 10).

Operand rules

- Specification of this operand is invalid if `0` is specified for the `PDCWAITTIME` operand of the client environment definition.
- Specify a value for this operand such that the combination of this value with the value of the `PDCWAITTIME` operand of the client environment definition results in a positive value.
- If the combination of the value of this operand with the value of the `PDCWAITTIME` operand of the client environment definition is `0` or smaller, `1` is assumed for the maximum wait time.

***maximum-wait-time-2-for-HiRDB-client-used-by-HiRDB-External-Data-Access-Adapter*:**

~ <unsigned integer>((0-65535)) <<600>> (minutes)

Specifies the maximum amount of time (units: seconds) the HiRDB client used by HiRDB External Data Access Adapter waits for a foreign HiRDB to respond after issuing a processing request. If maximum wait time 1 is omitted and `0` is specified for maximum time 2, the HiRDB client used by HiRDB External Data Access Adapter continues to wait for the foreign HiRDB to respond.

Operand specification methods

When specifying maximum wait time 2, do not forget the comma. The specification format follows:

```
PDCWAITTIME ,1200
```

■ Decision on specifying maximum wait time 1 or 2

If you specify the maximum wait time of the HiRDB client used by HiRDB External Data Access Adapter individually for each UAP, specify maximum wait time 1. In contrast, if you specify the same maximum wait time for the HiRDB client used by HiRDB External Data Access Adapter (the same value for all UAPs), specify maximum wait time 2.

■ Priority between maximum wait time 1 and 2

- If both maximum wait time 1 and 2 are simultaneously specified, maximum wait time 1 takes precedence.
- If this operand is omitted, the default value (`0`) of maximum wait time 1 is assumed.
- If the foreign server is HiRDB Version 06-01 or older, specification of maximum wait time 1 is invalid. Therefore, if both maximum wait time

1 and 2 are simultaneously specified, maximum wait time 2 is applied. If this operand is omitted, the default value (`600`) of maximum wait time 2 is assumed. If maximum wait time 1 only is specified by mistake, the default value (`600`) of maximum wait time 2 is assumed.

34) PDSWAITTIME [*maximum-wait-time-1-for-foreign-HiRDB*][,*maximum-wait-time-2-for-foreign-HiRDB*]

The function of this operand is the same as that of the `PDSWAITTIME` operand of the client environment definition. For details on the `PDSWAITTIME` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

maximum-wait-time-1-for-foreign-HiRDB:

~ <unsigned integer>((0-65534)) <<0>> (seconds)

Specifies the number of seconds to be added to the value of the `PDSWAITTIME` operand of the client environment definition. This operand determines the maximum amount of time a foreign HiRDB waits for the next request after it returns a response to the HiRDB client used by HiRDB External Data Access Adapter.

Operand specification methods

The maximum wait time is determined as follows:

Maximum wait time = *value-of-this-operand* + *value-of-PDSWAITTIME-operand-of-client-environment-definition*

For example, if `10` and `100` are specified for this operand and the `PDSWAITTIME` operand of the client environment definition, respectively, the maximum wait time is 110 seconds (10 + 100).

Operand rules

- Specification of this operand is invalid if `0` is specified for the `PDSWAITTIME` operand of the client environment definition.
- If the combination of the value of this operand with the value of the `PDSWAITTIME` operand of the client environment definition exceeds 65535, `65535` is assumed for the maximum wait time for the foreign HiRDB.

maximum-wait-time-2-for-foreign-HiRDB:

~ <unsigned integer>((0-65535)) <<600>> (minutes)

Specifies the maximum amount of time (units: seconds) a foreign HiRDB waits for the next request after it returns a response to the HiRDB client used by HiRDB External Data Access Adapter. If maximum wait time 1 is omitted and `0` is specified for maximum time 2, the foreign HiRDB continues to wait until the HiRDB client used by HiRDB External Data Access Adapter responds.

Operand specification methods

When specifying maximum wait time 2, do not forget the comma. The specification format follows:

```
PDSWAITTIME ,1200
```

■ Decision on specifying maximum wait time 1 or 2

If you specify the maximum wait time of the foreign HiRDB individually for each UAP, specify maximum wait time 1. In contrast, if you specify the same maximum wait time for the foreign HiRDB (the same value for all UAPs), specify maximum wait time 2.

■ Priority between maximum wait time 1 and 2

- If both maximum wait time 1 and 2 are simultaneously specified, maximum wait time 1 takes precedence.
- If this operand is omitted, the default value (`0`) of maximum wait time 1 is assumed.
- If the foreign server is HiRDB Version 06-01 or older, specification of maximum wait time 1 is invalid. Therefore, if both maximum wait time 1 and 2 are simultaneously specified, maximum wait time 2 is applied. If this operand is omitted, the default value (`600`) of maximum wait time 2 is assumed. If maximum wait time 1 only is specified by mistake, the default value (`600`) of maximum wait time 2 is assumed.

10.2.10 Operands related to troubleshooting information

35) PDCLTPATH *SQL-trace-file-and-error-log-file-output-destination-directory*

～ <path-name><<$PDDIR/spool/hubspool/*foreign-server-name*/>>

Specifies the output destination directory for the SQL trace file and error log file. For details on the SQL trace file and error log file, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

36) PDSQLTRACE *SQL-trace-file-size*

～ <unsigned integer>((0, 4096-2000000000)) (Bytes)

Specifies the size of the SQL trace file (units: bytes). The SQL trace file is output to the directory specified by the `PDCLTPATH` operand. If the `PDCLTPATH` operand is omitted, the SQL trace file is output to `$PDDIR/spool/hubspool/`*foreign-serve-name*`/`.

Operand rules

- If 0 is specified, the maximum file size is used.
- If this operand is omitted, an SQL trace is not output.

The function and specification method of this operand are the same as those of the PDSQLTRACE operand of the client environment definition. For details on the PDSQLTRACE operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

37) PDSQLTEXTSIZE *size-of-SQL-statements-to-be-output-to-SQL-trace-file*

~ ((4096-2000000))<<4096>>(bytes)

Specifies in bytes the size of the SQL statements to be output to the SQL trace file.

If this operand is omitted when an access path is acquired, 2000000 is assumed.

38) PDSQLEXECTIME YES | NO

Specifies whether to output SQL execution time to the SQL trace.

YES: SQL execution time is output to the SQL trace.

NO: SQL execution time is not output to the SQL trace.

Notes

- If YES is specified for this operand, SQL execution time is output to a UAP statistical report in microseconds.
- An execution time exceeding 24 hours cannot be correctly output to the SQL trace.

39) PDUAPERLOG *error-log-file-size*

~ <unsigned integer>((0, 4096-65536)) <<4096>> (Bytes)

Specifies the error log file size (units: bytes). The error log file is output to the directory specified by the PDCLTPATH operand. If the PDCLTPATH operand is omitted, the error log file is output to $PDDIR/spool/hubspool/*foreign-serve-name*/.

Operand rule

- If 0 is specified, the maximum file size is used.

The function and specification method of this operand are the same as those of the PDUAPERLOG operand of the client environment definition. For details on the PDUAPERLOG operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

40) PDPRMTRC YES | <u>NO</u>

Specifies whether to output the ? parameter in the SQL trace.

`YES`: Outputs the ? parameter in the SQL trace.

`NO`: Does not output the ? parameter in the SQL trace.

41) PDPRMTRCSIZE *maximum-data-size-of-?-parameter-information-inside-SQL-trace*

~ <unsigned integer>((4-4096)) <<256>> (Bytes)

Specifies the maximum data size for the ? parameter information to be output in the SQL trace.

Condition

The `PDPRMTRC` operand must be set to `YES`.

42) PDTRCMODE <u>ERR</u> | NONE

Specifies whether to output troubleshooting information (information with the file name `pderr`*XXXX*`.trc`) other than the SQL trace.

`ERR`: Outputs troubleshooting information.

`NONE`: Does not output troubleshooting information.

Troubleshooting information is output to the directory specified by the `PDCLTPATH` operand. If the `PDCLTPATH` operand is omitted, troubleshooting information is output to `$PDDIR/spool/hubspool/`*foreign-serve-name*`/`.

43) PDTRCPATH *dynamic-SQL-trace-file-output-destination-directory*

~ <pathname>

Specifies the output destination directory for a dynamic SQL trace file. You must specify this operand if you use the trace collection command (`pdtrcmgr`) to collect a dynamic SQL trace file.

If you specify in `pdtrcmgr` the directory specified in this operand, SQL trace files are output to the directory specified in this operand during subsequent connections.

For details on the trace collection command (`pdtrcmgr`), see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

44) PDUAPREPLVL [s][u][p][r] | a

Specifies the output information type for the UAP statistical report. For details on the UAP statistical report facility, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

s: Outputs information for each SQL statement. Also outputs SQL trace.

u: Outputs information for each UAP.

p: Outputs access path information.

r: Outputs intermediate information during SQL execution.

a: Outputs all information.

Condition

The PDCLTPATH operand must be specified.

Operand rules

- If this operand is omitted, only the SQL trace is output.
- You can specify s, u, p, and r as a combination of one or more letters. If you specify supr, it is the same as specifying a. If any letter other than s or a is specified, no SQL trace is output.
- If you specify only PDUAPREPLVL and not s, u, p, r, or a, an error occurs.

The function and specification method of this operand are the same as those of the PDUAPREPLVL operand of the client environment definition. For details on the PDUAPREPLVL operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

45) PDREPPATH *UAP-statistical-report-output-destination-directory*

～ <pathname>

Normally, the UAP statistical report is output to the directory specified by the PDCLTPATH operand. To output the UAP statistical report to a different directory, specify this operand.

A file is created for each connection or disconnection under a file name pd*HHMMSSmmm_xxx*_*(1 or 2).trc.

HHMMSSmmm: Connection time

xxx: Connection serial number

The function and specification method of this operand are the same as those of the PDREPPATH operand of the client environment definition. For details on the PDREPPATH operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

46) PDSQLTRCOPENMODE CNCT | SQL

Specifies the opening mode for the SQL trace file when collecting an SQL trace.

`CNCT`:

Outputs an SQL trace by opening or closing the SQL trace file for each connection or disconnection. Because this option results in smaller overhead than when each SQL statement is specified, it reduces the SQL trace output time. However, because information is written while the SQL trace is kept open, the SQL trace may not be correctly output if normal disconnection does not occur.

`SQL`:

Outputs an SQL trace by opening or closing the SQL trace file for each SQL statement (operation).

Condition

The `PDREPPATH` operand must be specified.

47) PDVWOPTMODE *access-path-information-acquisition-mode*

~ <unsigned integer>((0-2)) <<0>>

Specifies whether to collect access path information. The access path information collected here is used when executing the access path display utility. The access path information is output to `$PDDIR/spool/pdsqldump` of the foreign HiRDB (the unit containing a single server or front-end server).

`0`:

Does not collect access path information.

`1`:

Collects access path information. However, the information of an SQL statement for which the SQL object is inside a buffer is not output.

`2`:

Collects access path information. Even for an SQL statement with an SQL object inside a buffer, the SQL object is re-created and SQL information is output.

The function and specification method of this operand are the same as those of the `PDVWOPTMODE` operand of the client environment definition. For details on the `PDVWOPTMODE` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

10.2.11 Operands related to lock

48) PDLOCKLIMIT *upper-limit-for-lock-requests-to-foreign-HiRDB*

~ <unsigned integer>((0-32767)) <<0>>

Specifies the upper limit for lock requests to each foreign HiRDB. If `0` is specified or if this operand is omitted, no upper limit for lock requests is checked.

Specification guidelines

The number of locked resources differs for each SQL statement. Estimate the number of locked resources and use this estimate in computing the value to be specified for this operand. For determining the number of locked resources, see *E. Determining the Number of Locked Resources*.

For details on locks, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

10.2.12 Operands related to inner replica facility

49) PDDBACCS *RDAREA-generation-number-of-foreign-HiRDB*

~ <unsigned integer>((0-10))

Specify this operand if the foreign HiRDB uses the inner replica facility. For accessing an RDAREA that is not the current RDAREA inside an inner replica group, specify the generation number of that RDAREA. Specification of `0` means the original RDAREA.

This operand is applied to all inner replica groups defined in the foreign HiRDB. If the replica RDAREA of the generation specified by this operand is not defined, the current RDAREA inside the applicable inner replica group becomes the processing target. Therefore, when setting up a test environment that uses a replica RDAREA, do not mistakenly access the RDAREA for actual use. Make sure that replica RDAREAs of the specified generation are defined for all RDAREAs that are to be accessed.

50) PDDBORGUAP YES | <u>NO</u>

Specifies whether a UAP can be executed for a foreign HiRDB's original RDAREA that is in the online reorganization shutdown state.

`YES`:

A database operation (batch processing) by a UAP can be executed for the foreign HiRDB's original RDAREA.

`NO`:

A database operation by a UAP cannot be executed for the foreign HiRDB's original RDAREA.

10.2.13 Operands related to SQL optimization

51) PDSQLOPTLVL
***SQL-optimization-option-for-foreign-HiRDB*[,*SQL-optimization-option-for-foreign-HiRDB*]...**

~ <*identifier* or unsigned integer>

Specifies the SQL optimization option for a foreign HiRDB. For details on SQL optimization, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

If this operand is omitted, the SQL optimization option that is set by the foreign HiRDB is applied. Specify this operand when the SQL optimization option of the foreign HiRDB is not to be used.

Operand rule

The maximum length of the character string that can be specified for this operand is 575 bytes.

The function and specification method of this operand are the same as those of the `PDSQLOPTLVL` operand of the client environment definition. For details on the `PDSQLOPTLVL` operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

52) PDADDITIONALOPTLVL
***SQL-extension-optimization-option-for-foreign-HiRDB*[,*SQL-extension-optimization-option-for-foreign-HiRDB*]...**

~ <*identifier* or unsigned integer>

Specifies the SQL extension optimization option for a foreign HiRDB. For details on SQL extension optimization, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

If this operand is omitted, the SQL extension optimization option that was set by the foreign HiRDB is applied. Specify this operand when the SQL extension optimization option of the foreign HiRDB is not to be used.

Operand rule

The maximum length of the character string that can be specified for this operand is 575 bytes.

The function and specification method of this operand are the same as those of the PDADDITIONALOPTLVL operand of the client environment definition. For details on the PDADDITIONALOPTLVL operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

53) PDAGGR *number-of-groups-generated-for-foreign-HiRDB-grouping*

32-bit mode: ~ **<unsigned integer> ((0-30000000)) <<1024>>**

64-bit mode: ~ **<unsigned integer> ((0-2147483647)) <<1024>>**

Specifies the maximum number of groups to be generated in each server to determine the amount of memory to be used for a GROUP BY process of a foreign HiRDB. This operand becomes valid when rapid grouping processing, which is an SQL optimization option, is to be executed in the foreign HiRDB.

The function and specification method of this operand are the same as those of the PDAGGR operand of the client environment definition. For details on the PDAGGR operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

54) PDHASHTBLSIZE *hash-table-size-for-hash-join-subquery-hash-execution-by-foreign-HiRDB*

32-bit mode: ~ **<unsigned integer> ((128-524288)) (KB)**

64-bit mode: ~ **<unsigned integer> ((128-2097152)) (KB)**

Specifies a hash table size (units: KB) when hash join, subquery hash execution, which is an SQL optimization option, is to be executed in a foreign HiRDB. If this operand is omitted, the value of the pd_hash_table_size operand of the foreign HiRDB is assumed.

The function and specification method of this operand are the same as those of the PDHASHTBLSIZE operand of the client environment definition. For details on the PDHASHTBLSIZE operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

10.2.14 Operands related to the facility for output of extended SQL error information

55) PDUAPEXERLOGUSE YES | <u>NO</u>

Specifies whether a foreign HiRDB uses the facility for output of extended SQL error information.

YES: The facility for output of extended SQL error information is used.

NO: The facility for output of extended SQL error information is not used.

The function and specification method of this operand are the same as those of the PDUAPEXERLOGUSE operand in the client environment definition. For details about the PDUAPEXERLOGUSE operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

56) PDUAPEXERLOGPRMSZ ***maximum-data-size-of-parameter-information-to-be-output-to-error-log-file-and-SQL-error-report-file***

～ <unsigned integer>((0-32008))<<0>>(bytes)

When a foreign HiRDB uses the facility for output of extended SQL error information, this operand specifies the maximum data size for the parameter information to be output to an error log file.

- When 1 or a value greater than 1 is specified

  Parameter information is output to an error log file and an SQL error report file.

- When 0 is specified

  Parameter information is not output to an error log file or an SQL error report file.

If the parameter information is in the variable-length character string type, BLOB type, or BINARY type, the data size area also is included in the specified value. If the size of the parameter information to be output to an error log file and an SQL error report file exceeds the value specified for PDUAPEXERLOGPRMSZ, only the parameter information that fits in the file size is output, and the remainder is discarded.

The function and specification method of this operand are the same as those of the PDUAPEXERLOGPRMSZ operand in the client environment definition. For details about the PDUAPEXERLOGPRMSZ operand, see the *HiRDB Version 8 UAP Development Guide*.

10.2.15 Operands related to the BES connection holding facility

For details about the BES connection holding facility, see the *HiRDB Version 8 System Operation Guide*.

57) PDBESCONHOLD YES | NO

Specifies whether a foreign HiRDB uses the BES connection holding facility. Note that specification of this operand is ignored if the foreign HiRDB is connected to a HiRDB/Single Server.

Y: The BES connection holding facility is used.

N: The BES connection holding facility is not used.

If this operand is omitted, the value of the pd_bes_connection_hold operand in the back-end server definition of the foreign HiRDB is used.

The function and specification method of this operand are the same as those of the PDBESCONHOLD operand. For details about the PDBESCONHOLD operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

58) PDBESCONHTI *back-end-server-connection-hold-time*

~ <unsigned integer>((0-3600))(seconds)

When a foreign HiRDB uses the BES connection holding facility, this operand specifies the BES connection holding period in seconds.

When the BES connection holding facility is used, HiRDB monitors the period between the termination of a transaction and the execution of the next transaction. If this period is within the specified value, the connection between the front-end server and the back-end server is maintained. However, if this period exceeds the specified value, the connection between the front-end server and the back-end server is terminated after the transaction is terminated.

If 0 is specified for this operand, the period is not monitored. The connection between the front-end server and the back-end server is terminated only when the connection between the front-end server and a client is terminated by SQL DISCONNECT (xa_close if the XA library is being used) or because the value of the PDCWAITTIME operand is exceeded.

The function and specification method of this operand are the same as those of the PDBESCONHTI operand in the client environment definition. For details about the PDBESCONHTI operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

10.2.16 Operand related to the system switchover facility

59) PDHATRNQUEUING NO

When a foreign HiRDB is using the transaction queuing facility, this operand specifies whether to cancel the transaction queuing facility. The function and specification method of this operand are the same as those of the PDHATRNQUEUING operand in the client environment definition. For details about the PDHATRNQUEUING operand, see the *HiRDB Version 8 UAP Development Guide*. If the HiRDB version number of the foreign server is different, see the manual for the correct version.

10.3 Notes

(1) Client environment definitions applicable to a foreign HiRDB

The following type of client environment definition is applied to a foreign HiRDB:

- `PDISLLVL`

If the foreign HiRDB version is 06-02 or newer, the values of the following client environment definitions are applied to a foreign HiRDB.

- `PDCLTAPNAME`
- `PDEXWARN`
- `PDDLKPRIO`
- `PDLOCKSKIP`
- `PDCWAITTIMEWRNPNT`

(2) Client environment definitions whose fixed values are applied to a foreign HiRDB

Fixed values of the following client environment definitions are applied to a foreign HiRDB:

- `PDSTJTRNOUT`: Fixed to `NO`
- `PDDFLNVAL`: Fixed to `NOUSE`
- `PDSWATCHTIME`: Fixed to `0`
- `PDAUTOCONNECT`: Fixed to `OFF`

(3) When the foreign HiRDB version is older than the local HiRDB version

If the foreign HiRDB version is older than the local HiRDB version, the operands specified in the foreign server information definition may not be applied to the foreign HiRDB. Whether they are applied depends on whether the specified operands are supported by the foreign HiRDB.

Chapter

11. Foreign Server Information Definition (When a Foreign Server Is XDM/RD E2)

This chapter explains the foreign server information definition when a foreign server is XDM/RD E2. Create this definition for using the HiRDB External Data Access facility.

11.1 Operand formats

The foreign server information definition defines the execution environment for the HiRDB External Data Access facility. Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *11.2 Operand explanations*.

■ To users who are creating a foreign server information definition for the first time

First, determine the values to be specified for the operands shown in bold. Basically, once you specify these operands, you can use the HiRDB External Data Access facility.

■ Operands you can modify before restarting HiRDB

You can modify all operands in the foreign server information definition before restarting HiRDB (following planned termination, forced termination, or abnormal termination).

No.	Format	Operand category
1	[set pd_hb_db_con = STARTUP | ACCESS]	Connection to and disconnection from a foreign server
2	[set pd_hb_db_dis_con = SHUTDOWN | COMMIT]	
3	[set pd_hb_e_mode = ALL | NONE]	Foreign server error information
4	[set pd_hb_e_code = SQLCODE[,SQLCODE]...]	
5	[set pd_hb_e_size = *foreign-server-error-information-file-size*]	

No.	Format	Operand category
6	`[set pd_hb_l_path =` *foreign-server-interface-trace-information-file-output-destination-directory*`]`	Foreign server interface trace information
7	`[set pd_hb_l_size =` *foreign-server-interface-trace-information-file-size*`]`	
8	`[set pd_hb_l_prm = Y | N ]`	
9	`[set pd_hb_l_prm_size =` *maximum-data-size-of-?-parameter-information-to-be-output-to foreign-server-interface-trace-information-file*`]`	
10	`[set pd_hb_l_mode = ALL | PARTIAL]`	
11	**putenv PDCLTRDNODE** *XDM/RD-E2-database-identifier*	Connection to a foreign server
12	**putenv PDHOST** *DB-connection-server-host-name*[,*secondary-system-host-name*]	
13	**putenv PDNAMEPORT** *DB-connection-server-port-number*	
14	`[putenv PDCWAITTIME` `[`*maximum-wait-time-1-for-HiRDB-client-used-by-HiRDB-External-Data-Access-Adapter*`][,`*maximum-wait-time-2-for-HiRDB-client-used-by-HiRDB-External-Data-Access-Adapter*`]]`	System monitoring
15	`[putenv PDCLTPATH` *SQL-trace-file-and-error-log-file-output-destination-directory*`]`	Troubleshooting information
16	`[putenv PDSQLTRACE` *SQL-trace-file-size*`]`	
17	`[putenv PDUAPERLOG` *error-log-file-size*`]`	
18	`[putenv PDPRMTRC YES | NO ]`	
19	`[putenv PDPRMTRCSIZE` *maximum-data-size-of-?-parameter-information-inside-SQL-trace*`]`	
20	`[putenv PDBLKF` *number-of-block-transfer-rows-of-search-result-to-be-sent-from-XDM/RD-E2*`]`	Data transfer

11.2 Operand explanations

11.2.1 Operands related to connection to and disconnection from a foreign server

1) pd_hb_db_con = STARTUP | ACCESS

Specifies the timing to connect to a foreign server.

`STARTUP`:

Connects to a foreign server when a back-end server process is started.

When `STARTUP` is specified, the foreign server is connected even when there is no processing involving it. Therefore, if there is a limit on the number of foreign servers that can be connected, specify `ACCESS`.

`ACCESS`:

Connects to a foreign server only when a back-end server process requests access to the foreign server.

2) pd_hb_db_dis_con = SHUTDOWN | COMMIT

Specifies the timing to disconnect from a foreign server.

`SHUTDOWN`:

Disconnects from the foreign server when a back-end server process is terminated.

When `SHUTDOWN` is specified, the foreign server is connected even when there is no processing involving it. Therefore, if there is a limit on the number of foreign servers that can be connected, specify `COMMIT`.

`COMMIT`:

Disconnects from the foreign server immediately after `COMMIT` is issued to the foreign server.

11.2.2 Operands related to foreign server error information

3) pd_hb_e_mode = ALL | NONE

Specifies whether to output foreign server error information. For details on foreign server error information, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

`ALL`: Outputs foreign server error information.

`NONE`: Does not output foreign server error information.

4) pd_hb_e_code = *SQLCODE*[,*SQLCODE*]...

~ <signed integer>((-214748364 to -1))

For this operand, specify the foreign server's SQLCODE (value returned to SQLCODE of the SQL Communications Area SQLCA). The error message corresponding to the SQLCODE specified here is not output to the foreign server error information.

If this operand is omitted, all error messages corresponding to all SQLCODEs are output to the foreign server error information file. For details on SQLCODEs, see the documentation for each DBMS.

Operand rules

- When specifying multiple SQLCODEs, delimit them using commas.
- The maximum number of SQLCODEs that can be specified is 10.

5) pd_hb_e_size = *foreign-server-error-information-file-size*

~ <unsigned integer>((0, 4-4096))<<32>> (KB)

Specifies the size of the foreign server error information file (units: KB). To not limit the size, specify `0`.

11.2.3 Operands related to foreign server interface trace information

6) pd_hb_l_path = *foreign-server-interface-trace-information-file-output-destination-directory*

~ <pathname> <<$PDDIR/spool/>>

Specifies the output destination directory for foreign server interface trace information. Foreign server interface trace information is output to this directory. Two output files are created under this directory. Their names are `pdhl`*foreign-server-name-1* and `pdhl`*foreign-server-name-2*.

For details on foreign server interface trace information, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

7) pd_hb_l_size = *foreign-server-interface-trace-information-file-size*

~ <unsigned integer>((0, 4-4096)) <<1024>>(KB)

Specifies the size of the foreign server interface trace information file (units: KB). To not limit the size, specify `0`.

If the file size exceeds the value specified for this operand, the files are swapped and trace information is output to the other file.

Specification guideline

On the average, 300 bytes of information is output per trace (excluding the ?

parameter information).

Notes

- For a UAP executed by specifying the PDSQLTRACE operand of the client environment definition, foreign server interface trace information is collected for each foreign server. Because multiple transactions write data into a single file, a lock is applied during a writing operation. Therefore, collecting foreign server interface trace information lowers both transaction processing performance and concurrent executability.
- Depending on the access status of the foreign server, the file size may exceed the value specified for this operand. Therefore, collect foreign server interface trace information onto a disk with ample free space.

8) pd_hb_l_prm = Y | N

Specifies whether to output ? parameter information in foreign server interface trace information.

Y: Outputs ? parameter information.

N: Does not output ? parameter information.

9) pd_hb_l_prm_size = *maximum-data-size-of-?-parameter-information-to-be-output-to foreign-server-interface-trace-information-file*

~ <unsigned integer>((4-4096)) <<256>>(Bytes)

Specifies the maximum data size (units: bytes) of the ? parameter information to be output in the foreign server interface trace information file.

Condition

pd_hb_l_prm = Y must be specified.

10) pd_hb_l_mode = ALL | PARTIAL

Specifies whether to output all foreign server interface trace information.

ALL: Outputs all information.

PARTIAL: Outputs the information on only the facilities that communicate with the foreign server.

11.2.4 Operands related to connection to a foreign server

11) PDCLTRDNODE *XDM/RD-E2-database-identifier*

~ <identifier>

This operand cannot be omitted and must be specified.

Specifies the database identifier of the XDM/RD E2 to be connected. A database identifier means an RD-node specified in an XDM subsystem definition.

12) PDHOST *DB-connection-server-host-name*[,*secondary-system-host-name*]

~ <host name>

This operand cannot be omitted and must be specified.

Specifies the host name of the DB connection server to be connected. The host name can also be specified as an IP address. Specify for the IP address a decimal number that is delimited by a period after each byte, as shown as follows.

IP address format: *NNN.OOO.PPP.QQQ*

Normally, the host with the name that is specified first is connected. If this host cannot be connected, the host in the secondary system is connected.

13) PDNAMEPORT *DB-connection-server-port-number*

~ <unsigned integer>((5001-65535))

This operand cannot be omitted and must be specified.

Specifies the port number of the DB connection server to be connected.

For the port number, specify the server schedule number specified by the `SCHEDULENO` operand of the control space startup control statement and server space startup control statement of the DB connection server. If the `SCHEDULENO` operand is omitted, `40179` is assumed for the server schedule number (port number).

11.2.5 Operands related to system monitoring

14) PDCWAITTIME [*maximum-wait-time-1-for-HiRDB-client-used-by-HiRDB-External-Data-Access-Adapter*][,*maximum-wait-time-2-for-HiRDB-client-used-by-HiRDB-External-Data-Access-Adapter*]

The function of this operand is the same as that of the `PDCWAITTIME` operand of the client environment definition. For details on the `PDCWAITTIME` operand, see the manual *HiRDB Version 8 XDM/RD E2 Connection Facility*.

Specification of maximum wait time 1 for this operand is invalid. Be sure to specify only maximum wait time 2. If this operand is omitted, the default value (`600`) of maximum wait time 2 is assumed.

maximum-wait-time-1-for-HiRDB-client-used-by-HiRDB-External-Data-Access-Adapter:

~ <signed integer>((-65534 to 0))<<0>> (seconds)

Specifies the number of seconds to be deducted from the value of the

PDCWAITTIME operand of the client environment definition. This operand determines the maximum amount of time the HiRDB client used by HiRDB External Data Access Adapter waits for a foreign HiRDB to respond following a processing request.

Operand specification methods

The maximum wait time is determined as follows:

Maximum wait time = *value-of-this-operand* + *value-of-PDCWAITTIME-operand-of-client-environment-definition*

For example, if -10 and 100 are specified for this operand and the PDCWAITTIME operand of the client environment definition, respectively, the maximum wait time is 90 seconds (100 - 10).

Operand rules

- Specification of this operand is invalid if 0 is specified for the PDCWAITTIME operand of the client environment definition.
- Specify a value for this operand such that the combination of this value with the value of the PDCWAITTIME operand of the client environment definition results in a positive value.
- If the combination of the value of this operand with the value of the PDCWAITTIME operand of the client environment definition is 0 or smaller, 1 is assumed for the maximum wait time.

maximum-wait-time-2-for-HiRDB-client-used-by-HiRDB-External-Data-Access-Adapter:

~ <unsigned integer>((0-65535)) <<600>> (seconds)

Specifies the maximum amount of time (units: seconds) the HiRDB client used by HiRDB External Data Access Adapter waits for a foreign HiRDB (XDM/RD E2) to respond after issuing a processing request. If 0 is specified for maximum time 2, the HiRDB client used by HiRDB External Data Access Adapter continues to wait for the foreign HiRDB to respond.

Operand specification methods

When specifying maximum wait time 2, do not forget the comma. The specification format follows:

```
PDCWAITTIME,1200
```

11.2.6 Operands related to troubleshooting information

15) PDCLTPATH *SQL-trace-file-and-error-log-file-output-destination-directory*

~ <pathname><<$PDDIR/spool/hubspool/foreign-server-name/>>

Specifies the output destination directory for the SQL trace file and error log file. For details on the SQL trace file and error log file, see the *HiRDB Version 8 UAP Development Guide*.

16) PDSQLTRACE *SQL-trace-file-size*

～ <unsigned integer>((0,4096-2000000000)) (Bytes)

Specifies the size of the SQL trace file (units: bytes). The SQL trace file is output to the directory specified by the `PDCLTPATH` operand. If the `PDCLTPATH` operand is omitted, the SQL trace file is output to `$PDDIR/spool/hubspool/`*foreign-serve-name*`/`.

For details on the SQL trace facility, see the manual *HiRDB Version 8 XDM/RD E2 Connection Facility*.

Operand rules

- If `0` is specified, the maximum file size is used.
- If this operand is omitted, SQL trace is not output.

The function of this operand is the same as that of the `PDSQLTRACE` operand of the client environment definition. For details on the `PDSQLTRACE` operand, see the manual *HiRDB Version 8 XDM/RD E2 Connection Facility*.

17) PDUAPERLOG *error-log-file-size*

～ <unsigned integer>((0,4096-65536)) <<4096>> (Bytes)

Specifies the error log file size (units: bytes). The error log file is output to the directory specified by the `PDCLTPATH` operand. If the `PDCLTPATH` operand is omitted, the error log file is output to `$PDDIR/spool/hubspool/`*foreign-serve-name*`/`.

Operand rule

If `0` is specified, the maximum file size is used.

18) PDPRMTRC YES | <u>NO</u>

Specifies whether to output the ? parameter in the SQL trace.

`YES`: Outputs the ? parameter in the SQL trace.

`NO`: Does not output the ? parameter in the SQL trace.

19) PDPRMTRCSIZE *maximum-data-size-of-?-parameter-information-inside-SQL-trace*

～ <unsigned integer>((4-4096)) <<256>> (Bytes)

Specifies the maximum data size for the ? parameter information to be output in the SQL trace.

Condition

The PDPRMTRC operand must be set to YES.

11.2.7 Operands related to data transfer

20) PDBLKF
number-of-block-transfer-rows-of-search-result-to-be-sent-from-XDM/RD-E2

～ <unsigned integer>((1-4096)) <<10>>

Specifies the number of search result rows to be sent at a time from XDM/RD E2 to the DB access adapter.

The function of this operand is the same as that of the PDBLKF operand of the client environment definition. For details on the PDBLKF operand, see the manual *HiRDB Version 8 XDM/RD E2 Connection Facility*.

11.3 Client environment definitions with fixed values that are applied to a foreign server (XDM/RD E2)

The fixed value of the following client environment definition is applied to a foreign server (XDM/RD E2):

- `PDSRVTYP`: Fixed to `VOS3`

Chapter

12. Foreign Server Information Definition (When a Foreign Server Is ORACLE)

This chapter explains the foreign server information definition when a foreign server is ORACLE. Create this definition for using the HiRDB External Data Access facility.

12.1 Operand formats

The foreign server information definition defines the execution environment for the HiRDB External Data Access facility. Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *12.2 Operand explanations*. Note that in addition to the foreign server information definition, you also need an ORACLE client definition. For details, see the ORACLE documentation.

■ To users who are creating a foreign server information definition for the first time

First, determine the values to be specified for the operands shown in bold. Basically, once you specify these operands, you can use the HiRDB External Data Access facility.

■ Operands you can modify before restarting HiRDB

You can modify all operands in the foreign server information definition before restarting HiRDB (following planned termination, forced termination, or abnormal termination).

No.	Format	Operand category
1	`[set pd_hb_db_con = STARTUP	ACCESS]`	Connection to and disconnection from a foreign server
2	`[set pd_hb_db_dis_con = SHUTDOWN	COMMIT]`	
3	`[set pd_hb_e_mode = ALL	NONE]`	Foreign server error information
4	`[set pd_hb_e_code = SQLCODE[,SQLCODE]...]`	
5	`[set pd_hb_e_size =` *foreign-server-error-information-file-size*`]`	

No.	Format	Operand category
6	`[set pd_hb_l_path =` *foreign-server-interface-trace-information-file-output-destination-directory*`]`	Foreign server interface trace information
7	`[set pd_hb_l_size =` *foreign-server-interface-trace-information-file-size*`]`	
8	`[set pd_hb_l_prm = Y | N ]`	
9	`[set pd_hb_l_prm_size =` *maximum-data-size-of-?-parameter-information-to-be-output-to foreign-server-interface-trace-information-file*`]`	
10	`[set pd_hb_l_mode = ALL | PARTIAL]`	
11	`[set pd_hb_ary_fec_num =` *number of rows-to-be-fetched-at-once-with-FETCH-facility-that-uses-array*`]`	FETCH
12	`[set pd_hb_sql_trace = Y | N ]`	Troubleshooting information
13	`[set pd_hb_get_lock = Y | N ]`	Lock
14	`[set pd_hb_use_describe = Y | N ]`	DESCRIBE statement
15	`putenv PDNETSERVICE` *net-service-name-of-database-to-be-connected*	Connection to a foreign server

12.2 Operand explanations

12.2.1 Operands related to connection to and disconnection from a foreign server

1) pd_hb_db_con = STARTUP | ACCESS

Specifies the timing to connect to a foreign server.

`STARTUP`:

Connects to a foreign server when a back-end server process is started.

When `STARTUP` is specified, the foreign server is connected even when there is no processing involving it. Therefore, if there is a limit on the number of foreign servers that can be connected, specify `ACCESS`.

`ACCESS`:

Connects to a foreign server only when a back-end server process requests access to the foreign server.

2) pd_hb_db_dis_con = SHUTDOWN | COMMIT

Specifies the timing to disconnect from a foreign server.

`SHUTDOWN`:

Disconnects from the foreign server when a back-end server process is terminated.

When `SHUTDOWN` is specified, the foreign server is connected even when there is no processing involving it. Therefore, if there is a limit on the number of foreign servers that can be connected, specify `COMMIT`.

`COMMIT`:

Disconnects from the foreign server immediately after `COMMIT` is issued to the foreign server.

12.2.2 Operands related to foreign server error information

3) pd_hb_e_mode = ALL | NONE

Specifies whether to output foreign server error information. For details on foreign server error information, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

`ALL`: Outputs foreign server error information.

`NONE`: Does not output foreign server error information.

4) pd_hb_e_code = *SQLCODE*[,*SQLCODE*]...

~ <signed integer>((-214748364 to -1))

For this operand, specify the foreign server's SQLCODE (value returned to SQLCODE of the SQL Communications Area SQLCA). The error message corresponding to the SQLCODE specified here is not output to the foreign server error information.

If this operand is omitted, all error messages corresponding to all SQLCODEs are output to the foreign server error information file. For details on SQLCODEs, see the documentation for each DBMS.

Operand rules

- When specifying multiple SQLCODEs, delimit them using commas.
- The maximum number of SQLCODEs that can be specified is 10.

5) pd_hb_e_size = *foreign-server-error-information-file-size*

~ <unsigned integer>((0, 4-4096))<<32>> (KB)

Specifies the size of the foreign server error information file (units: KB). To not limit the size, specify `0`.

12.2.3 Operands related to foreign server interface trace information

6) pd_hb_l_path = *foreign-server-interface-trace-information-file-output-destination-directory*

~ <pathname> <<$PDDIR/spool/>>

Specifies the output destination directory for foreign server interface trace information. Foreign server interface trace information is output to this directory. Two output files are created under this directory. Their names are `pdhl`*foreign-server-name-1* and `pdhl`*foreign-server-name-2*.

For details on foreign server interface trace information, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

7) pd_hb_l_size = *foreign-server-interface-trace-information-file-size*

~ <unsigned integer>((0, 4-4096)) <<1024>>(KB)

Specifies the size of the foreign server interface trace information file (units: KB). To not limit the size, specify `0`.

If the file size exceeds the value specified for this operand, the files are swapped and trace information is output to the other file.

Specification guideline

On average, 300 bytes of information is output per trace (excluding the ?

parameter information).

Notes

- For a UAP executed by specifying the PDSQLTRACE operand of the client environment definition, foreign server interface trace information is collected for each foreign server. Because multiple transactions write data into a single file, a lock is applied during a writing operation. Therefore, collecting foreign server interface trace information lowers both transaction processing performance and concurrent executability.
- Depending on the access status of the foreign server, the file size may exceed the value specified for this operand. Therefore, collect foreign server interface trace information onto a disk with ample free space.

8) pd_hb_l_prm = Y | <u>N</u>

Specifies whether to output ? parameter information in foreign server interface trace information.

Y: Outputs ? parameter information.

N: Does not output ? parameter information.

9) pd_hb_l_prm_size =
maximum-data-size-of-?-parameter-information-to-be-output-to foreign-server-interface-trace-information-file

~ <unsigned integer>(((4-4096)) <<256>>(Bytes)

Specifies the maximum data size (units: bytes) of the ? parameter information to be output in the foreign server interface trace information file.

Condition

pd_hb_l_prm = Y must be specified.

10) pd_hb_l_mode = <u>ALL</u> | PARTIAL

Specifies whether to output all foreign server interface trace information.

ALL: Outputs all information.

PARTIAL: Outputs the information on only the facilities that communicate with the foreign server.

12.2.4 Operands related to FETCH

11) pd_hb_ary_fec_num =
rows-to-be-fetched-at-once-with-FETCH-facility-that-uses-array

~ <unsigned integer>((1-30000))

Specifies the number of rows to be fetched at one time with FETCH that uses an

array.

If the number of search result rows from a foreign table can be predicted to some degree, specify this operand. The communication overhead between HiRDB and the foreign server can be expected to decrease. For the SQL execution time and the number of FETCH operations to a foreign table based on the array FETCH count to the foreign server, check the statistical information related to foreign server usage. For details on the statistical information related to foreign server usage, see the *Statistics analysis utility* in the manual *HiRDB Version 8 Command Reference*.

If this operand is omitted, HiRDB determines automatically a value that is optimized for the HiRDB processing.

Specification guidelines

- If the specification value is too small, the number of communications between HiRDB and the foreign server increases, resulting in a greater overhead.
- If the specification value is too large, too large a memory area is used, resulting in a memory resource shortage.
- If the number of search result rows from a foreign table varies depending on the job, Hitachi recommends that this operand be omitted. Whether omitting this operand is advantageous depends on whether one or more rows of data collected from the foreign table fit in the communication buffer size specified by the `pd_sql_send_buff_size` operand. If the data fits, HiRDB efficiently processes the data collected from the foreign table. If the data does not fit. HiRDB suppresses the memory used for accessing the foreign table.

12.2.5 Operands related to troubleshooting information

12) pd_hb_sql_trace = Y | <u>N</u>

Specifies whether to output the SQL trace information issued to a foreign server (ORACLE). The output destination is specified by the initialization parameter of ORACLE. For details on this parameter, see the ORACLE documentation.

`Y`: Outputs the SQL trace information.

`N`: Does not output the SQL trace information.

12.2.6 Operands related to lock

13) pd_hb_get_lock = Y | <u>N</u>

Specifies whether to lock data to be referenced by a search SQL statement. In ORACLE, a search SQL statement normally does not lock data to be referenced.

Consequently, if a transaction that updates another database by referencing an ORACLE database and a transaction that updates the ORACLE database by referencing another database are simultaneously executed, the continuity of these transactions cannot be guaranteed. In this case, lock the data read from ORACLE to guarantee the continuity of the transactions.

`Y`:

Locks the data referenced by a search SQL statement. A lock is applied when a `FOR UPDATE` clause is specified in a `SELECT` statement. If `Y` is specified, the locked data cannot be referenced from other transactions, and as a result, the concurrent execution performance of ORACLE declines.

`N`:

Does not lock the data referenced by a search SQL statement.

12.2.7 Operands related to DESCRIBE statement

14) pd_hb_use_describe = Y | N

Specifies whether to allow a foreign server (ORACLE) to issue a `DESCRIBE` statement when receiving data from that foreign server. If the foreign server is allowed to issue a `DESCRIBE` statement, the data type of the foreign server can be mapped to a larger number of columns in a foreign table. For details on data type correspondence when a foreign table is used, see the manual *HiRDB Version 8 SQL Reference*.

`Y`: Allows the issuance of a `DESCRIBE` statement.

`N`: Does not allow the issuance of a `DESCRIBE` statement.

Specification guidelines

- Specify `Y` if a `DESCRIBE` statement is needed for mapping a data type to HiRDB.
- When `Y` is specified for this operand, a `DESCRIBE` statement is internally issued for each SQL execution, and therefore the performance may decrease compared to a case in which `N` is specified.

12.2.8 Operands related to connection to a foreign server

15) PDNETSERVICE *net-service-name-of-database-to-be-connected*

～ <character string>

This operand cannot be omitted and must be specified.

Specifies the net service name of the foreign server (ORACLE) to be connected. Specify the server name specified in the `tnsnames.ora` file of ORACLE. For details, see the ORACLE documentation.

Chapter

13. Foreign Server Information Definition (When a Foreign Server Is DB2)

This chapter explains the foreign server information definition when a foreign server is DB2 (DB2 Universal Database for OS/390 Version 6). Create this definition for using the HiRDB External Data Access facility.

13.1 Operand formats

The foreign server information definition defines the execution environment for the HiRDB External Data Access facility. Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *13.2 Operand explanations*. Note that in addition to the foreign server information definition, you also need DB2 client definition. For details, see the DB2 documentation.

- To users who are creating a foreign server information definition for the first time

  First, determine the values to be specified for the operands shown in bold. Basically, once you specify these operands, you can use the HiRDB External Data Access facility.

- Operands you can modify before restarting HiRDB

  You can modify all operands in the foreign server information definition before restarting HiRDB (following planned termination, forced termination, or abnormal termination).

No.	Format	Operand category
1	`[set pd_hb_db_con = STARTUP	ACCESS]`	Connection to and disconnection from a foreign server
2	`[set pd_hb_db_dis_con = SHUTDOWN	COMMIT]`	
3	`[set pd_hb_e_mode = ALL	NONE]`	Foreign server error information
4	`[set pd_hb_e_code = SQLCODE[,SQLCODE]...]`	
5	`[set pd_hb_e_size =` *foreign-server-error-information-file-size*`]`	

No.	Format	Operand category
6	[set pd_hb_l_path = *foreign-server-interface-trace-information-file-output-destination-directory*]	Foreign server interface trace information
7	[set pd_hb_l_size = *foreign-server-interface-trace-information-file-size*]	
8	[set pd_hb_l_prm = Y | N]	
9	[set pd_hb_l_prm_size = *maximum-data-size-of-?-parameter-information-to-be-output-to foreign-server-interface-trace-information-file*]	
10	[set pd_hb_l_mode = ALL | PARTIAL]	
11	[set pd_hb_ary_fec_num = *rows-to-be-fetched-at-once-with-FETCH-facility-that-uses-array*]	FETCH
12	[set pd_hb_use_describe = Y | N]	DESCRIBE statement
13	putenv PDDBNAME *name-of-database-to-be-connected*	Connection to a foreign server

13.2 Operand explanations

13.2.1 Operands related to connection to and disconnection from a foreign server

1) pd_hb_db_con = STARTUP | <u>ACCESS</u>

Specifies the timing to connect to a foreign server.

`STARTUP`:

Connects to a foreign server when a back-end server process is started.

When `STARTUP` is specified, the foreign server is connected even when there is no processing involving it. Therefore, if there is a limit on the number of foreign servers that can be connected, specify `ACCESS`.

`ACCESS`:

Connects to a foreign server only when a back-end server process requests access to the foreign server.

2) pd_hb_db_dis_con = SHUTDOWN | <u>COMMIT</u>

Specifies the timing to disconnect from a foreign server.

`SHUTDOWN`:

Disconnects from the foreign server when a back-end server process is terminated.

When `SHUTDOWN` is specified, the foreign server is connected even when there is no processing involving it. Therefore, if there is a limit on the number of foreign servers that can be connected, specify `COMMIT`.

`COMMIT`:

Disconnects from the foreign server immediately after `COMMIT` is issued to the foreign server.

13.2.2 Operands related to foreign server error information

3) pd_hb_e_mode = <u>ALL</u> | NONE

Specifies whether to output foreign server error information. For details on foreign server error information, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

`ALL`: Outputs foreign server error information.

`NONE`: Does not output foreign server error information.

4) pd_hb_e_code = ***SQLCODE*****[,*****SQLCODE*****]...**

～ <signed integer>((-214748364 to -1))

For this operand, specify the foreign server's SQLCODE (value returned to SQLCODE of the SQL Communications Area SQLCA). The error message corresponding to the SQLCODE specified here is not output to the foreign server error information.

If this operand is omitted, all error messages corresponding to all SQLCODEs are output to the foreign server error information file. For details on SQLCODEs, see the documentation for each DBMS.

Operand rules

- When specifying multiple SQLCODEs, delimit them using commas.
- The maximum number of SQLCODEs that can be specified is 10.

5) pd_hb_e_size = ***foreign-server-error-information-file-size***

～ <unsigned integer>((0, 4-4096))<<32>> (KB)

Specifies the size of the foreign server error information file (units: KB). To not limit the size, specify `0`.

13.2.3 Operands related to foreign server interface trace information

6) pd_hb_l_path =
foreign-server-interface-trace-information-file-output-destination-directory

～ <pathname> <<$PDDIR/spool/>>

Specifies the output destination directory for foreign server interface trace information. Foreign server interface trace information is output to this directory. Two output files are created under this directory. Their names are `pdhl`*foreign-server-name-1* and `pdhl`*foreign-server-name-2*.

For details on foreign server interface trace information, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

7) pd_hb_l_size = ***foreign-server-interface-trace-information-file-size***

～ <unsigned integer>((0, 4-4096)) <<1024>>(KB)

Specifies the size of the foreign server interface trace information file (units: KB). To not limit the size, specify `0`.

If the file size exceeds the value specified for this operand, the files are swapped and trace information is output to the other file.

Specification guideline

On the average, 300 bytes of information is output per trace (excluding the ?

parameter information).

Notes

- For a UAP executed by specifying the `PDSQLTRACE` operand of the client environment definition, foreign server interface trace information is collected for each foreign server. Because multiple transactions write data into a single file, a lock is applied during a writing operation. Therefore, collecting foreign server interface trace information lowers both transaction processing performance and concurrent executability.
- Depending on the access status of the foreign server, the file size may exceed the value specified for this operand. Therefore, collect foreign server interface trace information onto a disk with ample free space.

8) pd_hb_l_prm = Y | N

Specifies whether to output ? parameter information in foreign server interface trace information.

`Y`: Outputs ? parameter information.

`N`: Does not output ? parameter information.

9) pd_hb_l_prm_size = ***maximum-data-size-of-?-parameter-information-to-be-output-to foreign-server-interface-trace-information-file***

~ <unsigned integer>((4-4096)) <<256>>(Bytes)

Specifies the maximum data size (units: bytes) of the ? parameter information to be output in the foreign server interface trace information file.

Condition

`pd_hb_l_prm = Y` must be specified.

10) pd_hb_l_mode = ALL | PARTIAL

Specifies whether to output all foreign server interface trace information.

`ALL`: Outputs all information.

`PARTIAL`: Outputs the information on only the facilities that communicate with the foreign server.

13.2.4 Operands related to FETCH

11) pd_hb_ary_fec_num = ***rows-to-be-fetched-at-once-with-FETCH-facility-that-uses-array***

~ <unsigned integer>((1-30000))

Specifies the number of rows to be fetched at one time with FETCH that uses an

array.

If the number of search result rows from a foreign table can be predicted to some degree, specify this operand. The communication overhead between HiRDB and the foreign server can be expected to decrease. For the SQL execution time and the number of FETCH operations to a foreign table based on the array FETCH count to the foreign server, check the statistical information related to foreign server usage. For details on the statistical information related to foreign server usage, see the *Statistics analysis utility* in the manual *HiRDB Version 8 Command Reference*.

If this operand is omitted, HiRDB determines automatically a value that is optimized for the HiRDB processing.

Specification guidelines

- If the specification value is too small, the number of communications between HiRDB and the foreign server increases, resulting in a greater overhead.
- If the specification value is too large, too large a memory area is used, resulting in a memory resource shortage.
- If the number of search result rows from a foreign table varies depending on the job, Hitachi recommends that this operand be omitted. Whether omitting this operand is advantageous depends on whether one or more rows of data collected from the foreign table fits in the communication buffer size specified by the `pd_sql_send_buff_size` operand. If the data fits, HiRDB efficiently processes the data collected from the foreign table. If the data does not fit. HiRDB suppresses the memory used for accessing the foreign table.

13.2.5 Operand related to DESCRIBE statement

12) pd_hb_use_describe = Y | N

Specifies whether to allow a foreign server (DB2) to issue a `DESCRIBE` statement when receiving data from that foreign server. If the foreign server is allowed to issue a `DESCRIBE` statement, the data type of the foreign server can be mapped to a larger number of columns in a foreign table. For details on data type correspondence when a foreign table is used, see the manual *HiRDB Version 8 SQL Reference*.

`Y`: Allows the issuance of a `DESCRIBE` statement.

`N`: Does not allow the issuance of a `DESCRIBE` statement.

Specification guidelines

- Specify `Y` if a `DESCRIBE` statement is needed for mapping a data type to HiRDB.
- When `Y` is specified for this operand, a `DESCRIBE` statement is internally issued for each SQL execution, and therefore the performance may decrease compared to a case in which `N` is specified.

13.2.6 Operands related to connection to a foreign server

13) PDDBNAME *name-of-database-to-be-connected*

~ <character string>

This operand cannot be omitted and must be specified.

Specifies the name of the database of the foreign server (DB2) to be connected. For details, see the DB2 documentation.

Chapter
14. Hub Optimization Information Definition

This chapter explains the Hub optimization information definition. Create this definition for using the HiRDB External Data Access facility.

14.1 Operand formats

The Hub optimization information definition defines whether to execute each syntax of an input query with a foreign server. This definition minimizes the functional differences between HiRDB and a foreign server. When the values specified in the Hub optimization information definition are modified, incompatible facilities produce invalid results. Furthermore, syntaxes not supported by the foreign server may cause errors in the foreign server. Therefore, Hitachi recommends that you use the values explained in *14.4 Recommended values for various foreign servers* without any modifications.

Note that the numbers in the following table correspond to the numbers assigned to the operands explained in *14.2 Operand explanations*.

- Operands you can modify before restarting HiRDB

  You can modify all operands in the Hub optimization information definition before restarting HiRDB (following planned termination, forced termination, or abnormal termination).

No.	Format
1	`[set pd_hub_opt_on_cnd = 1 - 2]`
2	`[set pd_hub_opt_joined_table = 1 - 3]`
3	`[set pd_hub_opt_set_func = 1 - 4]`
4	`[set pd_hub_opt_case = 1 - 2]`
5	`[set pd_hub_opt_like = 1 - 4]`
6	`[set pd_hub_opt_grouping = 1 - 4]`
7	`[set pd_hub_opt_data_len = 1 - 2147483647]`
8	`[set pd_hub_opt_abs = 1 - 3]`
9	`[set pd_hub_opt_date = 1 - 3]`
10	`[set pd_hub_opt_time = 1 - 2]`
11	`[set pd_hub_opt_digits = 1 - 2]`
12	`[set pd_hub_opt_length = 1 - 8]`
13	`[set pd_hub_opt_lower_upper_type = 1 - 8]`
14	`[set pd_hub_opt_mod_div_type = 1 - 3]`

No.	Format
15	`[set pd_hub_opt_substr = 1 - 3]`
16	`[set pd_hub_opt_num = 1 - 2]`
17	`[set pd_hub_opt_datetime = 1 - 2]`
18	`[set pd_hub_opt_datetime_op = 1 - 5]`
19	`[set pd_hub_opt_trailing_spc = 1 - 2]`
20	`[set pd_hub_opt_nullable = 1 - 2]`
21	`[set pd_hub_opt_use_zero_string = 1 - 2]`
22	`[set pd_hub_opt_nchar = 1 - 2]`
23	`[set pd_hub_opt_nest_scalar = 0 - 65535]`
24	`[set pd_hub_opt_float = 1 - 2]`
25	`[set pd_hub_opt_table_num = 0 - 65535]`
26	`[set pd_hub_opt_time_24hour = 1 - 2]`

14.2 Operand explanations

1) pd_hub_opt_on_cnd = 1 - 2 ~ <unsigned integer> <<2>>

Specifies whether a foreign server is used to execute an outer join (`LEFT OUTER JOIN`).

`1`: The foreign server executes an outer join regardless of the `ON` condition.

`2`: The foreign server executes an outer join in the following cases:

- The `ON` condition consists of `outer table column = inner table column` linked using `AND`.
- The `ON` condition consists of `outer table column = inner table column` and `inner table column = constant` linked using `AND`.

2) pd_hub_opt_joined_table = 1 - 3 ~ <unsigned integer> <<2>>

Specifies whether a foreign server is used to execute a table join.

`1`: The foreign server executes an outer join, inner join, and inner table join.

`2`: The foreign server does not execute an inner table join or a join that includes both an outer join and (inner) join.

`3`: In addition to Condition 2, the foreign server does not execute a join that includes both an outer join and a `WITH` clause.

3) pd_hub_opt_set_func = 1 - 4 ~ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute set functions.

`1`: The foreign server executes set functions.

`2`: The foreign server does not execute set function.

`3`: If the argument of a `DISTINCT` set function does not specify a column, the foreign server does not execute this set function.

`4`: If `COUNT` of an `ALL` set function, `AVG` of a `DISTINCT` set function, or two or more kinds of `DISTINCT` set functions are described as in single query, the foreign server does not execute these set functions.

4) pd_hub_opt_case = 1 - 2 ~ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute `CASE` expressions.

`1`: The foreign server executes `CASE` expressions.

`2`: The foreign server does not execute `CASE` expressions.

Since COALESCE is converted into a scalar function value, it is executed by the foreign server even when `2` is specified for the option.

5) pd_hub_opt_like = 1 - 4 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute a `LIKE` predicate, `NOT LIKE` predicate, `XLIKE` predicate, and `NOT XLIKE` predicate.

`1`: The foreign server executes these predicates.

`2`: The foreign server does not execute these predicates.

`3`: The foreign server does not execute a `LIKE` predicate or `NOT LIKE` predicate only.

`4`: The foreign server does not execute an `XLIKE` predicate or `NOT XLIKE` predicate only.

Note

If a `LIKE` predicate, `NOT LIKE` predicate, `XLIKE` predicate, or `NOT XLIKE` predicate for the character string-type, variable-size character string-type, mixed character string-type, or variable-size mixed character string-type data contains two-byte special characters, or if a `LIKE` predicate, `NOT LIKE` predicate, `XLIKE` predicate, or `NOT XLIKE` predicate for the national character string-type or variable-size national character string-type data contains one-byte special characters, do not execute these predicates using the foreign server.

6) pd_hub_opt_grouping = 1 - 4 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute grouping.

`1`: The foreign server executes grouping.

`2`: The foreign server does not execute grouping.

`3`: The foreign server does not execute a `HAVING` clause without `GROUP BY`.

`4`: If a column is not specified for a grouping column, the foreign server does not execute grouping.

7) pd_hub_opt_data_len = 1 - 2147483647 ～ <unsigned integer> <<255>>

If the data size of the result of a value expression of a fixed-size character type, variable-size character type, fixed-size national character type, variable-size national character type, fixed-size mixed character type, or variable-size mixed character type exceeds the value (units: bytes) specified by this operand, HiRDB processes the value expression.

However, if a ? parameter, embedded variable, SQL parameter, SQL variable, or `USER` is specified outside the value expression, the foreign server processes the value expression even when the actual size of the ? parameter, embedded variable, SQL parameter, SQL variable, or `USER` exceeds the value specified by this operand. Therefore, an error may occur.

8) pd_hub_opt_abs = 1 - 3 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute scalar functions.

1: The foreign server executes scalar functions.

2: The foreign server does not execute scalar functions.

3: If the day interval type or time interval type is specified for the argument, the foreign server does not execute scalar functions.

9) pd_hub_opt_date = 1 - 3 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute the DATE-type column specification, scalar function `DATE()`, and `CHAR()` when the date type is specified. Note however that the foreign server does not execute these operations in the following case:

- `DATE()` in which the character type or time stamp type is specified for the first argument and format is specified for the second argument

1: The foreign server executes `DATE()`.

2: The foreign server does not execute `DATE()`.

3: The foreign server does not execute `DATE()` if the integer type is specified for its argument. The foreign server does not execute the DATE-type column specification, either. The foreign server executes `CHAR()` in which the date type is specified for the argument.

10) pd_hub_opt_time = 1 - 2 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute the scalar function `TIME()` and `CHAR()` when the time type is specified. Note however that the foreign server does not execute these operations in the following case:

- `TIME()` in which the character type is specified for the first argument and format is specified for the second argument

1: The foreign server executes `TIME()`.

2: The foreign server does not execute `TIME()`.

11) pd_hub_opt_digits = 1 - 2 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute the scalar function `DIGITS()`.

1: The foreign server executes `DIGITS()`.

2: The foreign server does not execute `DIGITS()`.

12) pd_hub_opt_length = 1 - 8 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute the scalar function LENGTH(). However, if the argument is the time stamp type or binary type, the foreign server does not execute this scalar function.

1: The foreign server executes LENGTH().

2: The foreign server does not execute LENGTH().

3: The foreign server does not execute LENGTH() if the argument is the mixed character type or variable-size mixed character type.

4: The foreign server does not execute LENGTH() if the argument is not the character type (that is, if the argument is a number, date, time, day interval, or time interval).

5: The foreign server does not execute LENGTH() if the argument is the BLOB type.

6: The foreign server does not execute LENGTH() if the argument is not the mixed character type, variable-size mixed char type, or character type (that is, if the argument is a number, date, time, day interval, or time interval).

7: The foreign server does not execute LENGTH() if the argument is not the character type (that is, if the argument is a number, date, time, day interval, or time interval) or if it is the BLOB type.

8: The foreign server does not execute LENGTH() if the argument is the mixed character type, variable-size mixed char type, or BLOB type.

Notes

> If the foreign server executes LENGTH for a value expression for which the size defined in the foreign server is different from the size defined in a foreign table, the execution result from the foreign server differs from the result obtained when HiRDB executes LENGTH after receiving data. In this case, specify 2 for this operand.

13) pd_hub_opt_lower_upper_type = 1 - 8 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute scalar functions LOWER and UPPER.

1: The foreign server executes these scalar functions.

2: The foreign server does not execute these scalar functions.

3: The foreign server does not execute these scalar functions if their arguments are the fixed- or variable-size character type.

4: The foreign server does not execute these scalar functions if their arguments are the fixed- or variable-size national character type.

5: The foreign server does not execute these scalar functions if their arguments are the fixed- or variable-size mixed character type.

6: The foreign server does not execute these scalar functions if their arguments are the fixed- or variable-size character string type, or fixed- or variable-size national character type.

7: The foreign server does not execute these scalar functions if their arguments are the fixed- or variable-size national character type, or fixed- or variable-size mixed character string type.

8: The foreign server does not execute these scalar functions if their arguments are the fixed- or variable-size character string type, or fixed- or variable-size mixed character string type.

14) pd_hub_opt_mod_div_type = 1 - 3 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute the scalar function `MOD()`, set function AVG, and a division operation.

1: The foreign server executes the function and the operation.

2: The foreign server does not execute the function and the operation.

3: The foreign server does not execute the function and the operation in the following cases:

- `MOD()` with the argument of the DECIMAL type
- When the argument is a DECIMAL-type set function `AVG`
- Division whose operation result is the DECIMAL type

15) pd_hub_opt_substr = 1 - 3 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute the scalar function `SUBSTR()`.

1: The foreign server executes `SUBSTR()`.

2: The foreign server does not execute `SUBSTR()`.

3: The foreign server does not execute `SUBSTR()` if its argument is the mixed character string type or variable-size mixed character string type.

16) pd_hub_opt_num = 1 - 2 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute the scalar functions `DECIMAL`, `FLOAT`, and `INTEGER`, which convert the type to the numerical data type.

1: The foreign server executes these functions.

2: The foreign server does not execute these functions.

17) pd_hub_opt_datetime = 1 - 2 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute scalar functions that extract the date or time. The target functions are listed as follows. However, the foreign server does not execute any function with an argument of the time stamp type.

- DAY
- HOUR
- MINUTE
- MONTH
- SECOND
- YEAR

`1`: The foreign server executes these functions.

`2`: The foreign server does not execute these functions.

18) pd_hub_opt_datetime_op = 1 - 5 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to execute date and time operations.

`1`: The foreign server executes these operations.

`2`: The foreign server does not execute these operations.

`3`: The foreign server does not execute these operations if a value expression with a labeled duration includes the scalar function `ABS` or `MOD`.

`4`: In addition to Condition 3, the foreign server does not execute the following operations:

- Day interval ± day interval
- Day interval ± decimal constant
- Day interval ± DECIMAL
- Day interval × integer operation
- Day interval ÷ integer operation
- Time interval ± time interval
- Time interval ± decimal constant
- Time interval ± DECIMAL
- Time interval × integer operation
- Time interval ÷ integer operation
- Date data ± date data
- Time data ± time data

5: Only date data ± value expression {*YEARS*|*MONTHS*|*DAYS*} is executed by the foreign server. The time operation is not executed by the foreign server.

19) pd_hub_opt_trailing_spc = 1 - 2 ～ <unsigned integer> <<2>>

Specifies whether a foreign server is used to execute the variable-size character string type that includes or has a possibility of including a trailing space (comparison predicate, BETWEEN predicate, IN predicate, set function MIN or MAX, SELECT DISTINCT specification, set function with DISTINCT specification, GROUP BY and ORDER BY clause).

Note that the foreign server never executes a character string-type comparison (comparison predicates <, <=, >, and >=) that includes column specification in which DIFFERENT is specified for the column option COLLATING_SEQUENCE, BETWEEN predicate, set functions MIN and MAX, ORDER BY clause, comparison between character string-type constants, comparison between a character string-type constant and a variable, or comparison between variables.

1: The foreign server executes these types.

2: The foreign server does not execute a variable-size character string-type comparison that includes or has a possibility of including a trailing space. The following value expressions fall in this category. If you use a variable, make sure that the value specified for the variable does not include a trailing space.

1. A value expression that is a character string constant that includes a trailing space
2. A value expression that is not a concatenation operation and that includes a variable-size character string-type column in which YES is specified for the column option TRAILING_SPACE
3. A value expression that is a scalar function SUBSTR with a return value that is the variable-size character string-type
4. A value expression that is a concatenation operation with a return value that is the variable-size character string-type and in which the right-most primary of the concatenation operation satisfies Conditions 1 through 3
5. A value expression that is a concatenation operation with a return value that is the variable-size character string-type and in which the right-most primary of the concatenation operation is the fixed-size character string type (excluding cases in which the right-most primary is a scalar function CHAR or DIGITS)

20) pd_hub_opt_nullable = 1 - 2 ～ <unsigned integer> <<2>>

Specifies whether a foreign server is used to execute a concatenation operation that includes a column in which YES or NULLABLE is specified for the column option.

1: The foreign server executes the concatenation operation.

2: The foreign server does not execute the concatenation operation.

21) pd_hub_opt_use_zero_string = 1 - 2 ～ <unsigned integer> <<2>>

Specifies whether to use a foreign server to evaluate a value specification that includes or has a possibility of including a 0-byte character string. The evaluation targets are 0-byte character string constants, embedded variables for which the character string type is assumed for the data type, SQL variables, SQL parameters, and the ? parameter.

1: The foreign server evaluates value specifications.

2: The foreign server does not evaluate a value specification that includes or has a possibility of including a 0-byte character string in any of the following items.

- Predicate
- GROUP BY clause
- ORDER BY clause
- Selection expression (that also specifies a GROUP BY clause, ORDER BY clause, DISTINCT, set function, or HAVING clause)

If the foreign server treats a 0-byte character string as a null value, specify 2.

22) pd_hub_opt_nchar = 1 - 2 ～ <unsigned integer> <<2>>

Specifies whether the national character-type character code set of a foreign server is fixed to 2 bytes and whether the embedded character is a 2-byte space.

1: Fixed to 2 bytes and the embedded character is a 2-byte space.

2: Not fixed to 2 bytes or the embedded character is not a 2-byte space.

If 2 is specified, the foreign server does not execute a predicate or value expression that includes the national character type or variable-size national character type. If the national character-type character code set of the foreign server is not fixed to 2 bytes, specify 2.

23) pd_hub_opt_nest_scalar = 0 - 65535 ～ <unsigned integer> <<0>>

Specifies the maximum number of nests for a scalar operation or set function to be executed by a foreign server. If 0 is specified, the foreign server executes these operations or functions regardless of the number of nests. Here, the nests are counted differently from HiRDB, and the number of nests for a scalar operation or set function is 1 regardless of the type of operation.

24) pd_hub_opt_float = 1 - 2 ～ <unsigned integer> <<1>>

Specifies whether a foreign server to execute a value expression, predicate, GROUP BY clause, ORDER BY clause, or selection expression (and a GROUP BY clause, ORDER BY clause, DISTINCT, set function, or HAVING clause is specified) that includes the floating-point type be used. Note that processing of value specification is not executed by the foreign server.

1: The foreign server executes these operations.

2: The foreign server does not execute these operations.

If the precision of the floating-point type is different between the machine or DBMS of the foreign server and the local HiRDB, specify 2.

25) pd_hub_opt_table_num = 0 - 65535 ～ <unsigned integer> <<0>>

If the number of tables specified in a single SQL statement exceeds the value specified for this operand, HiRDB processes the tables. If this operand is omitted or 0 is specified for it, tables are processed by a foreign server regardless of their number.

26) pd_hub_opt_time_24hour = 1 - 2 ～ <unsigned integer> <<1>>

Specifies whether a foreign server is used to execute the TIME type.

1: The foreign server executes the TIME type.

2: The foreign server does not execute the TIME type.

If 24 hours, 00 minutes, 00 seconds is stored in the TIME-type column of a foreign DBMS, specify 2.

14.3 Notes

- If execution of an incompatible function by a foreign server is specified, an error occurs during an execution attempt by the foreign server. If execution of a function that has the same syntax but different specification by a foreign server is specified, that function is processed according to the specification of the foreign server.
- A foreign server does not execute the following types of functions:
  - Function invocation
  - System-defined scalar function
  - `IS_USER_CONTAINED_IN_HDS_GROUP` function
  - `HEX` function
  - `TIMESTAMP` function
  - `TIMESTAMP_FORMAT` function
  - `BIT_AND_TEST` function
  - `VARCHAR_FORMAT` function
  - `DATE` function with a date format specified
  - `TIME` function with a date format specified
  - `POSITION` function
  - Row value constructor having two or more row value constructor elements
  - `IN` predicate containing a row value constructor that does not specify a value, on the right side of the equation
  - `WRITE` specification
  - `CAST` specification
  - `GET_JAVA_STORED_ROUTINE_SOURCE` specification
  - `LIMIT` clause
  - Comparison predicates that include a column specification that requires conversion between character data and numerical data
  - Concatenation operations that include a column specification that requires conversion between character data and numerical data
  - Scalar functions that include a column specification that requires conversion between character data and numerical data

  - Window function
  - SIMILAR predicate

- If an SQL parameter, SQL variable, USER, CURRENT_DATE, CURRENT_TIME, or CURRENT_TIMESTAMP is specified for a selection expression, that selection expression is not executed by a foreign server.
- If an embedded variable, ? parameter, SQL parameter, SQL variable, USER, CURRENT_DATE, CURRENT_TIME, or CURRENT_TIMESTAMP is described in the four arithmetic operations, date operation, concatenation operation, scalar function, or CASE expression, the foreign server does not execute the processing of the four arithmetic operations, date operation, concatenation operation, scalar function, or CASE expression.
- The foreign server does not execute a value expression, predicate, GROUP BY clause, ORDER BY clause, or selection expression (and a GROUP BY clause, ORDER BY clause, DISTINCT, set function, or HAVING clause is specified) that includes the following data types:
  - Time stamp type
  - BINARY type
- If there is a query in which a view table, a WITH clause query name, or a derived table is specified for a FROM clause, and if a view table, WITH clause, or derived table that becomes an internally derived table is specified for a FROM clause of this view table, derived query expression inside a WITH clause, or derived table, the foreign server does not execute the first query specification.
- If a set function is specified in a view table or a selection expression for a derived query expression in a WITH clause, and if the query specification selection expression specifying this view table or WITH clause query name in a FROM clause does not contain the column specification derived from the set function, the foreign server does not execute that SQL statement.
- If no set function or GROUP BY clause is specified, and a query specification selection expression with HAVING specified does not include a column specification, the foreign server does not execute that SQL statement.
- If the same column is specified in a view table or in multiple selection expressions for a derived query expression in a WITH clause, and if a GROUP BY clause is specified in this view table or a query in which a WITH clause query name is specified for a FROM clause, the foreign server does not execute that SQL statement.
- If an INSERT, UPDATE, or DELETE statement contains a syntax that is not executed by the foreign server, an error results. For statements other than INSERT, UPDATE, and DELETE, HiRDB handles the syntax that is not executed by the foreign server.

- If an SQL parameter, SQL variable, USER, CURRENT_DATE, CURRENT_TIME, or CURRENT_TIMESTAMP is specified in the operand of a comparison predicate with no column specification, the foreign server does not execute that SQL statement.

14.4 Recommended values for various foreign servers

This section explains the recommended values of the operands for various types of foreign servers.

The values in the Hub optimization information definition sample file are the same as the values recommended here. Therefore, use the values in the Hub optimization information definition sample file without any modification. For the storage location of the Hub optimization information definition sample file, see the *HiRDB External Data Access Version 7 Description and User's Guide*.

Operand name	HiRDB Version 6 or later	HiRDB Version 5.0	XDM/RD E2	ORACLE	DB2
pd_hub_opt_on_cnd	1	1	1	2	1
pd_hub_opt_joined_table	1	3	1	2	1
pd_hub_opt_set_func	1	3	3	1	4
pd_hub_opt_case	1	1	1	2	1
pd_hub_opt_like[1]	1	1	4	4	4
pd_hub_opt_grouping	1	1	4	3	4
pd_hub_opt_data_len[2]	255	255	255	255	248
pd_hub_opt_abs	1	1	1	3	2
pd_hub_opt_date	1	1	1	3	1
pd_hub_opt_time	1	1	1	2	1
pd_hub_opt_digits	1	1	1	2	1
pd_hub_opt_length	1	1	3	7	3
pd_hub_opt_lower_upper_type	1	1	1	6	7
pd_hub_opt_mod_div_type	1	1	2	1	3
pd_hub_opt_substr	1	1	3	2	2
pd_hub_opt_num	1	1	1	2	1
pd_hub_opt_datetime	1	1	1	2	1
pd_hub_opt_datetime_op	1	1	1	5	4

Operand name	HiRDB Version 6 or later	HiRDB Version 5.0	XDM/RD E2	ORACLE	DB2
pd_hub_opt_trailing_spc	1	1	1	1	2
pd_hub_opt_nullable	1	1	1	2	1
pd_hub_opt_use_zero_string	1	1	1	2	1
pd_hub_opt_nchar[3]	1	1	2	2	1
pd_hub_opt_nest_scalar	0	0	0	0	100
pd_hub_opt_float	2	2	2	2	2
pd_hub_opt_table_num	64	64	64	64	15
pd_hub_opt_time_24hour	1	1	1	1	2

[1] If a LIKE predicate, NOT LIKE predicate, XLIKE predicate, or NOT XLIKE predicate for the character string-type, variable-size character string-type, mixed character string-type, or variable-size mixed character string-type data contains two-byte special characters, or if a LIKE predicate, NOT LIKE predicate, XLIKE predicate, or NOT XLIKE predicate for the national character string-type or variable-size national character string-type data contains one-byte special characters, do not execute these predicates using the foreign server.

[2] Although the maximum size of the character string type that can be defined in DB2 is 255 bytes, the maximum size of a character string-type constant that can be used is 248. If you need to use a character string-type constant that is 249 bytes or longer, use a concatenation operation.

[3] If the national character-type character code set of the foreign server is not fixed to 2 bytes, specify 2.

Appendixes

A. List of Operands

Table A-1 lists all the operands provided in the HiRDB system definitions; it also indicates whether or not each operand can be modified during a restart. The operands are listed in alphabetical order. Note that the operands for the following definitions are not described in this appendix. You can modify the operands for these definitions before restarting HiRDB (following planned termination, forced termination, or abnormal termination).

- UAP environment definition
- Foreign server information definition (HiRDB External Data Access facility)
- Hub optimization information definition (HiRDB External Data Access facility)

Table A-1: Operands defined in the HiRDB system definition and their modifiability during restart

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
HORCMINST	Y	—	—	—	—	—	—	Y	Y
pd_additional_optimize_level[7]	Y	—	—	—	Y	—	—	Y	Y
pd_alias_cache_size	—	—	Y	Y	Y	—	—	Y[2,3]	Y[2]
pd_assurance_index_no	Y	—	—	—	—	—	—	Y[4]	Y
pd_assurance_table_no	Y	—	—	—	—	—	—	Y[4]	Y
pd_audit	Y	Y	—	—	—	—	—	N	N
pd_aud_async_buff_count	Y	Y	—	—	—	—	—	Y	Y
pd_aud_async_buff_retry_intvl	Y	Y	—	—	—	—	—	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_aud_async_buff_size	Y	Y	—	—	—	—	—	Y	Y
pd_aud_file_name	Y	Y	—	—	—	—	—	Y	Y
pd_aud_file_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
pd_aud_max_generation_num	Y	Y	—	—	—	—	—	Y	Y
pd_aud_max_generation_size	Y	Y	—	—	—	—	—	Y	Y
pd_aud_no_standby_file_opr	Y	—	—	—	—	—	—	Y	Y
pd_audit_def_buffer_size	—	—	Y	Y	Y	—	—	Y	Y
pd_auth_cache_size	—	—	Y	Y	Y	—	—	Y[2,3]	Y[2]
pd_auto_vrup	Y	—	—	—	—	—	—	N	N
pd_bes_connection_hold	—	—	Y	—	—	—	Y	Y	Y
pd_bes_conn_hold_trn_interval	—	—	Y	—	—	—	Y	Y	Y
pd_bes_shmpool_size	—	—	Y	—	—	—	Y	Y[2]	Y[2]
pd_cancel_down_msgchange	Y	—	—	—	—	—	—	Y	Y
pd_cancel_dump	Y	Y	—	—	—	—	—	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_change_clt_ipaddr	Y	Y	—	—	—	—	—	Y	Y
pd_check_pending	Y	—	—	—	—	—	—	Y	Y
pd_client_waittime_over_abort	Y	—	—	—	—	—	—	Y	Y
pd_cmdhold_precheck	Y	—	—	—	—	—	—	Y	Y
pd_command_deadlock_priority	Y	—	—	—	—	—	—	Y	Y
pd_connect_errmsg_hide	Y	—	—	—	—	—	—	Y	Y
pd_constraint_name	Y	—	—	—	—	—	—	Y	Y
pd_cwaittime_report_dir	Y	Y	—	—	—	—	—	Y	Y
pd_cwaittime_report_size	Y	Y	—	—	—	—	—	Y	Y
pd_cwaittime_wrn_pnt[7]	Y	—	—	Y	Y	—	—	Y	Y
pd_dbbuff_attribute	Y	Y	—	—	—	—	—	Y	Y
pd_dbbuff_lock_interval	Y	—	—	—	—	—	—	Y	Y
pd_dbbuff_lock_release_detect	Y	—	—	—	—	—	—	Y	Y
pd_dbbuff_lock_spn_count	Y	—	—	—	—	—	—	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_dbbuff_lru_option	Y	—	—	—	—	—	—	Y	Y
pd_dbbuff_modify	Y	—	—	—	—	—	—	N	N
pd_dbbuff_rate_updpage	Y	—	—	—	—	—	—	Y	Y
pd_dbbuff_wait_interval	Y	Y	—	—	—	—	—	Y	Y
pd_dbbuff_wait_spn_count	Y	Y	—	—	—	—	—	Y	Y
pd_dbsync_altwrite_skip	Y	—	—	—	—	—	—	Y	Y
pd_dbsync_lck_release_count	—	—	Y	Y	—	Y	Y	Y	Y
pd_dbsync_point	Y	—	—	—	—	—	—	N	Y
pd_db_io_error_action	Y	Y	—	—	—	—	—	Y	Y
pd_deadlock_priority_use	Y	—	—	—	—	—	—	Y	Y
pd_debug_info_netstat	Y	—	—	—	—	—	—	Y	Y
pd_dec_sign_normalize	Y	—	—	—	—	—	—	N	Y
pd_def_buf_control_area_assign	Y	—	—	—	—	—	—	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_delete_reserved_word_file[7]	Y	—	—	—	—	—	—	Y	Y
pd_dfw_awt_process	—	—	Y	Y	—	Y	Y	Y	Y
pd_dfw_syncpoint_skip_limit	—	—	Y	Y	—	Y	Y	Y	Y
pd_dic_shmpool_size	—	—	Y	—	—	Y	—	Y[2]	Y[2]
pd_directory_server	Y	—	—	—	—	—	—	Y	Y
pd_down_watch_proc	Y	Y	—	—	—	—	—	Y	Y
pd_dump_suppress_watch_time	Y	Y	—	—	—	—	—	Y	Y
pd_fes_lck_pool_size	—	—	Y	—	Y	—	—	Y	Y
pd_floatable_bes	—	—	—	—	Y	—	—	Y	Y
pd_foreign_server_libpath	—	—	—	—	—	—	Y	Y	Y
pd_ha	Y	—	—	—	—	—	—	N	N
pd_ha_acttype	—	Y	—	—	—	—	—	N	N
pd_ha_agent	—	Y	—	—	—	—	—	N	N
pd_ha_ipaddr_inherit	Y	Y	—	—	—	—	—	N	N
pd_ha_max_act_guest_servers	—	Y	—	—	—	—	—	N	N

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_ha_max_server_process	—	Y	—	—	—	—	—	N	Y[2]
pd_ha_mgr_rerun	Y	—	—	—	—	—	—	Y	Y
pd_ha_prc_cleanup_check	Y	—	—	—	—	—	—	Y	Y
pd_ha_process_count	—	Y	—	—	—	—	—	Y	Y
pd_ha_resource_act_wait_time	Y	Y	—	—	—	—	—	Y	Y
pd_ha_restart_failure	—	Y	—	—	—	—	—	Y	Y
pd_ha_server_process_standby	—	Y	—	—	—	—	—	N	N
pd_ha_switch_timeout	Y	Y	—	—	—	—	—	Y	Y
pd_ha_transaction[7]	Y	—	—	—	—	—	—	Y	Y
pd_ha_trn_queuing_wait_time	Y	—	—	—	—	—	—	Y	Y
pd_ha_trn_restart_retry_time	Y	—	—	—	—	—	—	Y	Y
pd_ha_unit	—	Y	—	—	—	—	—	N	N
pd_hash_table_size[7]	Y	—	—	—	—	—	—	N	N
pd_hashjoin_hashing_mode[7]	Y	—	—	—	—	—	—	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_hostname	—	Y	—	—	—	—	—	N	N
pd_host_watch_interval	Y	—	—	—	—	—	—	Y	Y
pd_indexlock_mode	Y	—	—	—	—	—	—	N	Y
pd_inner_replica_control	Y	—	—	—	—	—	—	N	Y
pd_ipc_conn_count	Y	—	—	—	—	—	—	Y	Y
pd_ipc_conn_interval	Y	—	—	—	—	—	—	Y	Y
pd_ipc_conn_nblock	Y	—	—	—	—	—	—	Y	Y
pd_ipc_conn_nblock_time	Y	—	—	—	—	—	—	Y	Y
pd_ipc_inet_bufsize	Y	Y	—	—	—	—	—	Y	Y
pd_ipc_recv_count	Y	Y	—	—	—	—	—	Y	Y
pd_ipc_send_count	Y	Y	—	—	—	—	—	Y	Y
pd_ipc_send_retrycount	Y	Y	—	—	—	—	—	Y	Y
pd_ipc_send_retrysleeptime	Y	Y	—	—	—	—	—	Y	Y
pd_ipc_tcp_nodelayack	Y	—	—	—	—	—	—	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_ipc_unix_bufsize	Y	—	—	—	—	—	—	Y	Y
pd_java_archive_directory	Y	Y	—	—	—	—	—	Y	Y
pd_java_classpath	Y	Y	—	—	—	—	—	Y	Y
pd_java_libpath	Y	Y	—	—	—	—	—	Y	Y
pd_java_option	Y	—	—	—	—	—	—	Y	Y
pd_java_routine_stack_size	Y	—	—	—	—	—	—	Y	Y
pd_java_runtimepath	Y	Y	—	—	—	—	—	Y	Y
pd_java_stdout_file	Y	Y	Y	Y	Y	Y	Y	Y	Y
pd_jp1_event_level	Y	—	—	—	—	—	—	Y	Y
pd_jp1_event_msg_out	Y	—	—	—	—	—	—	Y	Y
pd_jp1_use	Y	—	—	—	—	—	—	N	N
pd_key_resource_type	Y	—	—	—	—	—	—	N	Y
pd_large_file_use	Y	—	—	—	—	—	—	N	Y
pd_lck_deadlock_info	Y	Y	—	—	—	—	—	Y	Y
pd_lck_hash_entry	—	—	Y	Y	Y	Y	Y	Y	Y
pd_lck_pool_size	—	—	Y	Y	—	Y	Y	Y[2]	Y[2]

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_lck_queue_limit	Y	—	—	—	—	—	—	Y	Y
pd_lck_release_detect	Y	Y	—	—	—	—	—	Y	Y
pd_lck_release_detect_interval	Y	Y	—	—	—	—	—	Y	Y
pd_lck_until_disconnect_cnt	—	—	Y	Y	—	Y	Y	N	Y
pd_lck_wait_timeout	Y	Y	—	—	—	—	—	Y	Y
pd_list_initialize_timing	Y	—	—	—	—	—	—	Y	Y
pd_log_auto_unload_path	—	—	—	Y	Y	Y	Y	N	N
pd_log_dual[6]	—	—	Y	Y	Y	Y	Y	N	N
pd_log_dual_write_method	—	—	Y	Y	Y	Y	Y	Y	Y
pd_log_max_data_size[6]	—	—	Y	Y	Y	Y	Y	Y[2,3]	Y[2]
pd_log_org_no_standby_file_opr	Y	—	—	—	—	—	—	Y	Y
pd_log_org_reflected_logpoint	Y	—	—	—	—	—	—	Y[5]	Y[5]
pd_log_rec_leng	—	—	Y	Y	Y	Y	Y	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_log_remain_space_check	—	—	Y	Y	Y	Y	Y	Y	Y
pd_log_rerun_reserved_file_open	—	—	Y	Y	Y	Y	Y	Y	Y
pd_log_rerun_swap	—	—	Y	Y	Y	Y	Y	Y	Y
pd_log_rpl_no_standby_file_opr	Y	—	—	—	—	—	—	Y	Y
pd_log_sdinterval	—	—	Y	Y	Y	Y	Y	Y	Y
pd_log_singleoperation	—	—	Y	Y	Y	Y	Y	N	N
pd_log_swap_timeout	—	—	Y	Y	Y	Y	Y	Y	Y
pd_log_unload_check	—	—	Y	Y	Y	Y	Y	Y	Y
pd_log_write_buff_count	—	—	Y	Y	Y	Y	Y	Y[3]	Y
pd_lv_mirror_use	Y	—	—	—	—	—	—	Y	Y
pd_master_file_name	Y	—	—	—	—	—	—	N	N
pd_max_access_tables	Y	—	—	—	—	—	—	Y[3]	Y
pd_max_access_tables_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
pd_max_add_dbbuff_no	—	—	Y	Y	—	Y	Y	N	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
`pd_max_add_dbbuff_shm_no`	—	—	Y	Y	—	Y	Y	N	Y
`pd_max_ard_process`	—	—	Y	Y	—	Y	Y	Y	Y
`pd_max_bes_process`	—	—	Y	—	—	—	Y	N	Y[2]
`pd_max_commit_write_reclaim_no`	Y	—	—	—	—	—	—	Y	Y
`pd_max_dic_process`	—	—	Y	—	—	Y	—	N	Y[2]
`pd_max_file_no`	Y	—	—	—	—	—	—	Y[2,3]	Y
`pd_max_file_no_wrn_pnt`	Y	—	—	—	—	—	—	Y	Y
`pd_max_foreign_server`	Y	—	—	—	—	—	—	N	Y
`pd_max_list_users`	Y	—	—	—	—	—	—	Y[3]	Y
`pd_max_list_count`	Y	—	—	—	—	—	—	N	Y
`pd_max_list_users_wrn_pnt`	Y	—	—	—	—	—	—	Y	Y
`pd_max_list_count_wrn_pnt`	Y	—	—	—	—	—	—	Y	Y
`pd_max_open_fds`	—	—	—	Y	—	—	Y	Y	Y
`pd_max_open_holdable_cursors`	—	—	Y	Y	—	Y	Y	N	N

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_max_rdarea_no	Y	—	—	—	—	—	—	N	Y
pd_max_rdarea_no_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
pd_max_recover_process	Y	Y	—	—	—	—	—	Y	Y
pd_max_reflect_process_count	Y	—	—	—	—	—	—	N	N
pd_max_server_process	Y	Y	—	—	—	—	—	Y	Y
pd_max_users	Y	—	—	—	—	—	—	N	Y
pd_max_users_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
pd_mlg_file_size	Y	—	—	—	—	—	—	Y	Y
pd_mlg_msg_log_unit	Y	—	—	—	—	—	—	N	N
pd_mode_conf	Y	—	—	—	—	—	—	Y	Y
pd_module_trace_max	Y	Y	Y	Y	Y	Y	Y	Y	Y
pd_module_trace_timer_level	Y	Y	Y	Y	Y	Y	Y	Y	Y
pd_name_port	Y	—	—	—	—	—	—	Comp	Comp
pd_node_name	—	—	—	Y	Y	—	—	Y	Y
pd_non_floatable_bes	—	—	—	—	Y	—	—	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_nowait_scan_option	Y	—	—	—	—	—	—	Y	Y
pd_oltp_holdcr	Y	—	—	—	—	—	—	Y	Y
pd_optimize_level[7]	Y	—	—	—	Y	—	—	Y	Y
pd_overflow_suppress	Y	—	—	—	—	—	—	Y	Y
pd_pageaccess_mode	Y	—	—	—	—	—	—	Y	Y
pd_plugin_ixmk_dir	—	—	—	Y	—	—	Y	N	Y
pd_process_count	—	—	Y	Y	Y	Y	Y	Y[2]	Y[2]
pd_process_terminator	Y	—	—	—	—	—	—	Y	Y
pd_process_terminator_max	Y	—	—	—	—	—	—	Y	Y
pd_queue_watch_time	Y	—	—	—	—	—	—	Y	Y
pd_queue_watch_timeover_action	Y	—	—	—	—	—	—	Y	Y
pd_rdarea_list_no_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
pd_rdarea_open_attribute	Y	—	—	—	—	—	—	Y	Y
pd_rdarea_open_attribute_use	Y	—	—	—	—	—	—	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_rdarea_warning_point	Y	—	—	—	—	—	—	Y	Y
pd_redo_allpage_put	Y	—	—	—	—	—	—	Y	Y
pd_reduced_check_time	Y	—	—	—	—	—	—	Y	Y
pd_registered_port	Y	Y	—	—	—	—	—	N	Comp
pd_registered_port_check	Y	Y	—	—	—	—	—	N	Comp
pd_registered_port_level	Y	Y	—	—	—	—	—	N	Y
pd_registry_cache_size	—	—	Y	Y	Y	—	—	Y	Y
pd_rise_disaster_mode	Y	—	—	—	—	—	—	Y	Y
pd_rise_fence_level	Y	—	—	—	—	—	—	Y	Y
pd_rise_pairvolume_combination	Y	—	—	—	—	—	—	N	N
pd_rise_use	Y	—	—	—	—	—	—	N	N
pd_rorg_predict	Y	—	—	—	—	—	—	Y	Y
pd_routine_def_cache_size	—	—	Y	Y	Y	—	—	Y	Y
pd_rpc_trace	Y	Y	Y	Y	Y	Y	Y	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_rpc_trace_name	Y	Y	Y	Y	Y	Y	Y	Y	Y
pd_rpc_trace_size	Y	Y	Y	Y	Y	Y	Y	Y	Y
pd_rpl_hdepath	—	Y	—	—	—	—	—	N	N
pd_rpl_init_start	Y	—	—	—	—	—	—	N	N
pd_rpl_reflect_mode	Y	—	—	—	—	—	—	N	Y
pd_sds_shmpool_size	—	—	Y	Y	—	—	—	Y[2]	Y[2]
pd_server_cleanup_interval	—	—	Y	Y	Y	Y	Y	Y	Y
pd_server_entry_queue	Y	Y	—	—	—	—	—	Y	Y
pd_service_port	Y	Y	—	—	—	—	—	Comp	Comp
pd_shared_rdarea_use	Y	—	—	—	—	—	—	N	Y
pd_shmpool_attribute	Y	Y	—	—	—	—	—	Y	Y
pd_space_level[7]	Y	—	—	—	—	—	—	Y	Y
pd_spd_assurance_count	—	—	Y	Y	Y	Y	Y	N	N
pd_spd_assurance_msg	—	—	Y	Y	Y	Y	Y	Y	Y
pd_spd_dual	—	—	Y	Y	Y	Y	Y	N	N

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_spd_max_data_size	—	—	Y	Y	Y	Y	Y	N	N
pd_spd_reduced_mode	—	—	Y	Y	Y	Y	Y	N	Y
pd_spd_reserved_file_auto_open	—	—	Y	Y	Y	Y	Y	N	Y
pd_spd_syncpoint_skip_limit	—	—	Y	Y	Y	Y	Y	Y	Y
pd_spool_cleanup	Y	Y	—	—	—	—	—	Y	Y
pd_spool_cleanup_interval	Y	Y	—	—	—	—	—	Y	Y
pd_spool_cleanup_interval_level	Y	Y	—	—	—	—	—	Y	Y
pd_spool_cleanup_level	Y	Y	—	—	—	—	—	Y	Y
pd_sql_object_cache_size	Y	—	Y	Y	Y	Y	Y	N	Y
pd_sql_send_buff_size	Y	—	—	—	—	—	—	Y	Y
pd_start_level	Y	—	—	—	—	—	—	Y	Y
pd_start_skip_unit	Y	—	—	—	—	—	—	Y	Y
pd_start_time_out	Y	—	—	—	—	—	—	Y	Y
pd_statistics	Y	—	—	—	—	—	—	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_stj_file_size	Y	Y	—	—	—	—	—	Y[3]	Y
pd_stj_buff_size	Y	Y	—	—	—	—	—	Y[3]	Y
pd_sts_file_name_1-7	—	—	—	Y	Y	Y	Y	N	N
pd_sts_initial_error	—	—	Y	Y	Y	Y	Y	Y	Y
pd_sts_last_active_file	—	—	—	Y	Y	Y	Y	Y	Y
pd_sts_last_active_side	—	—	—	Y	Y	Y	Y	Y	Y
pd_sts_last_active_side_sub	—	—	—	Y	Y	Y	Y	Y	Y
pd_sts_last_active_subfile	—	—	—	Y	Y	Y	Y	Y	Y
pd_sts_singleoperation	—	—	Y	Y	Y	Y	Y	Y	Y
pd_sts_subfile_name_1 to 7	—	—	—	Y	Y	Y	Y	N	N
pd_substr_length	Y	—	—	—	—	—	—	Y	Y
pd_svr_castoff_size	—	—	Y	Y	Y	Y	Y	Y	Y
pd_sysdef_default_option	Y	—	—	—	—	—	—	N	Y
pd_syssts_file_name_1-7	—	Y	—	—	—	—	—	N	N

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_syssts_initial_error	—	Y	—	—	—	—	—	Y	Y
pd_syssts_last_active_file	—	Y	—	—	—	—	—	Y	Y
pd_syssts_last_active_side	—	Y	—	—	—	—	—	Y	Y
pd_syssts_last_active_side_sub	—	Y	—	—	—	—	—	Y	Y
pd_syssts_last_active_subfile	—	Y	—	—	—	—	—	Y	Y
pd_syssts_singleoperation	—	Y	—	—	—	—	—	Y	Y
pd_syssts_subfile_name_1 to 7	—	Y	—	—	—	—	—	N	N
pd_system_complete_wait_time	Y	—	—	—	—	—	—	Y	Y
pd_system_dbsync_point	Y	—	—	—	—	—	—	N	Y
pd_system_id	Y	—	—	—	—	—	—	N	N
pd_table_def_cache_size	—	—	Y	Y	Y	—	—	Y[2,3]	Y[2]
pd_tcp_inet_bufsize	Y	Y	—	—	—	—	—	Y	Y
pd_tcp_unix_bufsize	Y	Y	—	—	—	—	—	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_term_watch_count	Y	Y	—	—	—	—	—	Y	Y
pd_thdlock_pipe_retry_interval	Y	Y	—	—	—	—	—	Y	Y
pd_thdlock_retry_time	Y	Y	—	—	—	—	—	Y	Y
pd_thdlock_sleep_func	Y	—	—	—	—	—	—	Y	Y
pd_thdlock_wakeup_lock	Y	Y	—	—	—	—	—	Y	Y
pd_thdspnlk_spn_count	Y	Y	—	—	—	—	—	Y	Y
pd_thred_max_stack_size	Y	—	—	—	—	—	—	Y	Y
pd_trn_commit_optimize	Y	—	—	—	—	—	—	Y	Y
pd_trn_rerun_branch_auto_decide	Y	—	—	—	—	—	—	Y	Y
pd_trn_send_decision_interval	Y	—	—	—	—	—	—	Y	Y
pd_trn_send_decision_intval_sec	Y	—	—	—	—	—	—	Y	Y
pd_trn_send_decision_retry_time	Y	—	—	—	—	—	—	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_trn_watch_time	Y	—	—	—	—	—	—	Y	Y
pd_type_def_cache_size	—	—	Y	Y	Y	—	—	Y	Y
pd_uap_exerror_log_dir	Y	Y	—	—	—	—	—	Y	Y
pd_uap_exerror_log_param_size[7]	Y	Y	—	—	—	—	—	Y	Y
pd_uap_exerror_log_size	Y	Y	—	—	—	—	—	Y	Y
pd_uap_exerror_log_use[7]	Y	—	—	Y	Y	—	—	Y	Y
pd_unit_id	—	Y	—	—	—	—	—	N	N
pd_utl_buff_size	Y	—	—	—	—	—	—	Y[2]	Y[2]
pd_utl_exec_mode	Y	—	—	—	—	—	—	N	Y
pd_utl_exec_time	Y	—	—	—	—	—	—	Y	Y
pd_view_def_cache_size	—	—	Y	Y	Y	—	—	Y[2,3]	Y[2]
pd_watch_pc_client_time[7]	—	—	Y	Y	Y	—	—	Y	Y
pd_watch_resource	Y	—	—	—	—	—	—	Y	Y
pd_watch_time	Y	Y	—	—	—	—	—	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pd_work_buff_expand_limit	—	—	Y	Y	—	Y	Y	Y	Y
pd_work_buff_mode	—	—	Y	Y	—	Y	Y	Y	Y
pd_work_buff_size	—	—	Y	Y	—	Y	Y	Y	Y
pd_work_table_option	Y	—	—	—	—	—	—	Y	Y
pdwork_wrn_pnt	Y	—	—	—	—	—	—	Y	Y
pdbuffer	Y	—	—	—	—	—	—	N	Y
pdcltgrp	Y	—	—	—	—	—	—	Y	Y
PDCONFPATH	—	Y	—	—	—	—	—	Comp	Comp
pdhagroup	Y	—	—	—	—	—	—	N	N
pdhibegin	Y	—	—	—	—	—	—	Y	Y
pdhubopt	—	—	—	—	Y	—	—	N	Y
pdlogadfg -d spd	—	—	—	Y	Y	Y	Y	Y[1]	Y[1]
pdlogadfg -d ssp	—	—	—	Y	Y	Y	Y	Y[1]	Y[1]
pdlogadfg -d sys[6]	—	—	—	Y	Y	Y	Y	Y[1]	Y[1]
pdlogadpf -d spd	—	—	—	Y	Y	Y	Y	Y[1]	Y[1]
pdlogadpf -d ssp	—	—	—	Y	Y	Y	Y	Y[1]	Y[1]
pdlogadpf -d sys[6]	—	—	—	Y	Y	Y	Y	Y[1]	Y[1]
pdmlgput	Y	—	—	—	—	—	—	Y	Y
pdplgprm	—	—	—	Y	Y	Y	Y	Y	Y

Operand	Definition type							Modifiable after forced or abnormal termination?	Modifiable after planned termination?
	SYS	UNT	SVR	SDS	FES	DS	BES		
pdplugin	Y	—	—	—	—	—	—	N	N
pdstart	Y	—	—	—	—	—	—	N	N
pdstbegin	Y	—	—	—	—	—	—	Y	Y
pdunit	Y	—	—	—	—	—	—	N	N
pdwork	—	—	—	Y	—	Y	Y	N	N
SHMMAX	Y	Y	—	—	—	—	—	Y	Y
TZ	Y	—	—	—	—	—	—	Y	Y

Y: Yes, specification value can be modified.

N: No, specification value cannot be modified.

Comp (compatibility): Operand can be modified as long as compatibility is maintained.

—: Not applicable.

SYS: System common definition

UNT: Unit control information definition

SVR: Server common definition

SDS: Single server definition

FES: Front-end server definition

DS: Dictionary server definition

BES: Back-end server definition

[1] Operands may be added, but not deleted or modified.

[2] If the specified value is too small, it may not be possible to restart HiRDB. When this is the case, restore the value to what it was before the change and then restart HiRDB.

[3] If the specified value is too large, it may not be possible to restart HiRDB. When this is the case, restore the value to what is was before the change and then restart HiRDB.

[4] If v6compatible or v7compatible is specified in the pd_sysdef_default_option operand, the value in effect before the change is used to start HiRDB even if the definition is changed during the HiRDB restart.

[5] Some restrictions apply to operand value modifications. For details, see the operand explanations.

[6] Care must be taken with respect to modifying this operand while HiRDB Datareplicator (extracted side) is linked. Before adding, changing, or deleting the specification of this operand in this case, first stop HiRDB Datareplicator. If the definition of this operand is changed while HiRDB Datareplicator is running, extraction by HiRDB Datareplicator may fail when HiRDB is restarted.

[7] The values for the following HiRDB system definition operands can be different for each client. To change a value for a client, specify the operand in the client environment definition. For details about the client environment definition, see the *HiRDB Version 8 UAP Development Guide*.

HiRDB system definition operand	Client environment definition operand
pd_additional_optimize_level	PDADDITIONALOPTLVL
pd_cwaittime_wrn_pnt	PDCWAITTIMEWRNPNT
pd_delete_reserved_word_file	PDDELRSVWDFILE
pd_ha_transaction	PDHATRNQUEUING
pd_hash_table_size	PDHASHTBLSIZE
pd_hashjoin_hashing_mode	PDHJHASHINGMODE
pd_optimize_level	PDSQLOPTLVL
pd_space_level	PDSPACELVL
pd_uap_exerror_log_param_size	PDUAPEXERLOGPRMSZ
pd_uap_exerror_log_use	PDUAPEXERLOGUSE
pd_watch_pc_client_time	PDSWATCHTIME

B. Operand Specification Values

This appendix provides an overview of what is specified in each of the HiRDB system definition operands. Note that the operands for the following definitions are not described in this appendix.

- Foreign server information definition (HiRDB External Data Access facility)
- Hub optimization information definition (HiRDB External Data Access facility)

(1) Operands related to system configuration

Operand explanation	Operand name
Specifies a HiRDB identifier.	`pd_system_id`
Specifies a HiRDB port number.	`pd_name_port`
Specifies the first HiRDB file in the master directory RDAREA.	`pd_master_file_name`
Allocates a unit to a host.	`pdunit`
Allocates a server to a host.	`pdstart`
Specifies a unit identifier.	`pd_unit_id`
Specifies a standard host name.	`pd_hostname`

(2) Operands related to maximum concurrent executions

Operand explanation	Operand name
Specifies the maximum number of concurrent connections.	`pd_max_users`
Specifies the maximum number of server processes that can be activated concurrently within a single unit.	`pd_max_server_process`
Specifies the maximum number of tables that can be accessed concurrently by a transaction.	`pd_max_access_tables`
Modifies the maximum number of utilities that can be executed concurrently.	`pd_utl_exec_mode`
Specifies the maximum number of `pdreclaim` commands (with the `-p` option specified) that can be executed concurrently.	`pd_max_commit_write_reclaim_no`

(3) Operands related to processes

Operand explanation	Operand name
Specifies the maximum number of processes that can be activated in any back-end server.	`pd_max_bes_process`
Specifies the maximum number of processes that can be activated in any dictionary server.	`pd_max_dic_process`
Specifies the number of resident processes that can be activated at server startup.	`pd_process_count`
Specifies in minutes the interval checking for nonresident server processes in HiRDB that are to be stopped. This facility is applied when the number of executing server processes exceeds the number of processes that can be made resident (value specified by the `pd_process_count` operand).	`pd_server_cleanup_interval`
Specifies the maximum amount of memory space that can be used by a server process. If the amount of memory being used by a server process exceeds the value specified here, that server process is terminated.	`pd_svr_castoff_size`
Specifies the maximum number of files and pipes accessed by HiRDB processes.	`pd_max_open_fds`
Specifies the number of processes necessary for asynchronous READ operations when using the asynchronous READ facility. For a HiRDB/Parallel Server, this operand specifies the number of processes per server (back-end server or dictionary server).	`pd_max_ard_process`
Specifies the number of processes to be processed in parallel when the facility for parallel writes in deferred write processing is used.	`pd_dfw_awt_process`

(4) Operands related to work tables

Operand explanation	Operand name
Specifies the method of allocating buffers for creating work tables for temporary storage of information during SQL execution.	`pd_work_buff_mode`
Specifies the size of the work table buffer to be used for SQL execution.	`pd_work_buff_size`
Specifies the upper limit for expanding the work table buffer when the space in this buffer becomes insufficient.	`pd_work_buff_expand_limit`
Specifies the names of HiRDB file system areas for work table files.	`pdwork`

(5) Operands related to HiRDB startup

Operand explanation	Operand name
Specifies the HiRDB system startup method.	`pd_mode_conf`
Specifies the amount of time to wait for completion of the `pdstart` command execution.	`pd_system_complete_wait_time`
Specifies the maximum amount of time to wait for completion of HiRDB start preparation.	`pd_start_time_out`
Specifies the maximum number of times in a row that abnormal termination can occur during HiRDB restart processing.	`pd_term_watch_count`

(6) Operands related to reduced activation

Operand explanation	Operand name
Specifies whether or not reduced activation is to be used.	`pd_start_level`
Specifies the amount of time to wait for receipt of a notice of reduced activation startup.	`pd_reduced_check_time`
Specifies the names of units that need not be started because of errors, etc., during HiRDB startup.	`pd_start_skip_unit`

(7) Operands related to HiRDB processing

Operand explanation	Operand name
Specifies the timing for committing database changes to a file.	`pd_dbsync_point`
Specifies the timing for committing to file updates in the following types of RDAREAs: • Master directory RDAREA • Data directory RDAREA • Data dictionary RDAREAs • Data dictionary LOB RDAREAs • Registry RDAREAs • Registry LOB RDAREAs	`pd_system_dbsync_point`
When an update buffer reference request is issued during synchronization point acquisition processing, the transaction process that issued the reference request handles the writing of the update buffer contents into the database. This operand specifies whether or not the handling of this write processing is to be skipped.	`pd_dbsync_altwrite_skip`
Specifies whether or not errors are to be suppressed during operation.	`pd_overflow_suppress`

Operand explanation	Operand name
Specifies whether to start the post-processing process, which executes post-processing when a HiRDB process is abnormally terminated, when starting HiRDB.	`pd_process_terminator`
Specifies the number of post-processing processes to be started when starting HiRDB.	`pd_process_terminator_max`
Specifies a space conversion level when the space configuration facility is used.	`pd_space_level`
Specifies whether or not the facility for conversion to a DECIMAL signed normalized number is to be used.	`pd_dec_sign_normalize`
If contention occurs in the HiRDB server process during concurrent execution of UAPs, processing requests may sometimes be temporarily queued. This operand specifies what HiRDB must do in this case.	`pd_server_entry_queue`
Specifies the type of sleep function (for waiting for a specified amount of time) to be used when acquiring a lock between threads.	`pd_thdlock_sleep_func`
Specifies a thread lock release notification method.	`pd_thdlock_wakeup_lock`
Specifies in microseconds the interval at which to check for thread lock release.	`pd_thdlock_pipe_retry_interval`
Specifies in microseconds the thread lock sleep time.	`pd_thdlock_retry_time`
Specifies a spin count for thread spin lock.	`pd_thdspnlk_spn_count`
Specifies the page access mode to be used for database retrieval.	`pd_pageaccess_mode`
Specifies the method to be used for checking for RDAREA hold.	`pd_cmdhold_precheck`
Specifies the processing to be performed by HiRDB when an input/ output error occurs in an RDAREA (excluding the master directory RDAREA).	`pd_db_io_error_action`
Specifies whether to hide the error cause in the message that is output when a connection attempt fails.	`pd_connect_errmsg_hide`
Specifies whether or not the error messages output when a server process is terminated forcibly are to be changed.	`pd_cancel_down_msgchange`

(8) Operands related to full recovery processing

Operand explanation	Operand name
Specifies the number of processes to be recovered during full recovery processing. For a HiRDB/Parallel Server, this operand specifies the number of processes to be recovered per server (dictionary server or back-end server).	`pd_max_recover_process`
Specifies whether to output to a database the pages updated after a synchronization point during full recovery processing.	`pd_redo_allpage_put`

(9) Operands related to transaction decision processing

Operand explanation	Operand name
Specifies whether or not an undecided transaction that has branched from a transaction is to be decided automatically when a unit is restarted. An undecided transaction occurs when a unit terminates abnormally while the latter transaction's first phase of `COMMIT` is incomplete.	`pd_trn_rerun_branch_auto_decide`
Specifies the interval (in seconds) for sending an automatic decision instruction to a branched transaction when the previous send operation has failed for some reason.	`pd_trn_send_decision_intval_sec`
Specifies the interval (in minutes) for sending an automatic decision instruction to a branched transaction when the previous send operation has failed for some reason.	`pd_trn_send_decision_interval`
Specifies the maximum amount of time to wait for a decision completion notice to be returned after an automatic decision instruction has been sent to a branched transaction.	`pd_trn_send_decision_retry_time`
Specifies the maximum amount of time to wait for receiving communication (prepare, commit instruction, or response) between transaction branches during transaction synchronization point processing executed in the HiRDB server process.	`pd_trn_watch_time`
Specifies whether to use one-phase commit in a HiRDB/Parallel Server's commitment control.	`pd_trn_commit_optimize`

(10) Operands related to SQL optimization

Operand explanation	Operand name
Specifies SQL optimization options.	`pd_optimize_level`
Specifies SQL extension optimizing options.	`pd_additional_optimize_level`
Specifies the hashing method to be used when `hash join, subquery hash execution` is specified for the SQL extension optimizing option.	`pd_hashjoin_hashing_mode`

Operand explanation	Operand name
Specifies the size of the hash table to be used when "application of hash-execution of a hash join or a subquery" is specified as an SQL optimization option.	`pd_hash_table_size`
Specifies one of the following HiRDB processing methods to be used for executing an SQL statement that uses a work table: • Lock acquisition method when AND multiple indexes are used • Suppression of message output during automatic extension of the work table buffer	`pd_work_table_option`
Specifies back-end servers that can be used as floating servers.	`pd_floatable_bes`
Specifies back-end servers that are not to be used as floating servers.	`pd_non_floatable_bes`

(11) Operands related to the facility for output of extended SQL error information

Operand explanation	Operand name
Specifies whether to use the facility for output of extended SQL error information.	`pd_uap_exerror_log_use`
Specifies the directory in which to store SQL error report files.	`pd_uap_exerror_log_dir`
Specifies the maximum size of an SQL error report file.	`pd_uap_exerror_log_size`
Specifies the maximum data size for the parameter information to be output to an error log file and an SQL error report file.	`pd_uap_exerror_log_param_size`

(12) Operands related to the SQL reserved word deletion facility

Operand explanation	Operand name
Specifies the name of an SQL reserved word deletion file when the SQL reserved word deletion facility is used.	`pd_delete_reserved_word_file`

(13) Operands related to narrowed retrieval

Operand explanation	Operand name
Specifies the maximum number of users who can own lists concurrently.	`pd_max_list_users`
Specifies the maximum number of lists that one user can create.	`pd_max_list_count`
Specifies the list initialization (deletion) timing. Normally, lists are initialized when HiRDB is started (including restart). You can use this operand to change the initialization timing.	`pd_list_initialize_timing`

(14) Operands related to system monitoring

Operand explanation	Operand name
Specifies the monitoring time (units: minutes) for monitoring the execution time of the following utilities: • Database load utility (pdload command) • Database reorganization utility (pdrorg command) If the execution of a utility is not terminated within the monitoring time specified by this operand, the executing utility is forcibly terminated, and the error information for identifying the cause of no response is output.	pd_utl_exec_time
Specifies a maximum execution time for SQLs that are executed in a HiRDB server process. If execution of an SQL statement is not completed within the specified amount of time, execution of that SQL statement is terminated.	pd_watch_time
If messages cannot be extracted from the message queue within the time specified by this operand, a warning message (KFPS00888-W or KFPS00889-E) is output (the message queue monitoring facility).	pd_queue_watch_time
Specifies the processing to be performed by HiRDB when messages cannot be extracted from the message queue within the time specified by the pd_queue_watch_time operand.	pd_queue_watch_timeover_action
If abnormal terminations of server processes exceed the value specified in this operand, HiRDB (applicable unit for a HiRDB/ Parallel Server) is abnormally terminated (the facility for monitoring abnormal process terminations).	pd_down_watch_proc
Specifies the interval at which the other host's operation status is to be checked.	pd_host_watch_interval
Specifies the maximum amount of time to wait for the next request from a HiRDB client after the HiRDB server returns a response to a request from a Windows-compatible HiRDB client.	pd_watch_pc_client_time
Specifies the number of successive synchronization point dumps that can be skipped (number of skips in a single transaction).	pd_spd_syncpoint_skip_limit
Specifies the maximum number of times synchronization point dumps can be skipped (number of skips per transaction) when synchronization point dump acquisition is delayed by deferred write processing.	pd_dfw_syncpoint_skip_limit
Specifies whether or not a warning message related to resource usage is to be issued.	pd_watch_resource
Specifies whether to output the warning message when the number of concurrent connections reaches or exceeds the specified percentage of the maximum value specified by the pd_max_users operand.	pd_max_users_wrn_pnt

Operand explanation	Operand name
Specifies whether or not a warning message related to the number of concurrently accessible base tables specified by the pd_max_access_tables operand is to be issued.	pd_max_access_tables_wrn_pnt
Specifies whether to output the warning message when the number of RDAREAs reaches or exceeds the specified percentage of the maximum value specified by the pd_max_rdarea_no operand.	pd_max_rdarea_no_wrn_pnt
Specifies whether to output the warning message when the number of HiRDB files comprising an RDAREA reaches or exceeds the specified percentage of the maximum value specified by the pd_max_file_no operand.	pd_max_file_no_wrn_pnt
Specifies whether or not a warning message related to the HiRDB file system areas for work table files specified by the pdwork operand is to be issued.	pdwork_wrn_pnt
Specifies as a percentage of the number of users who can create lists (as specified by the pd_max_list_users operand) the point at which the number of users actually using lists is to cause a warning message to be issued.	pd_max_list_users_wrn_pnt
Specifies as a percentage of the number of lists that can be created per user (as specified by the pd_max_list_count operand) the point at which the number of lists created by a user is to cause a warning message to be issued.	pd_max_list_count_wrn_pnt
Specifies as a percentage of the number of lists that can be created at a server the point at which a warning message is to be issued.	pd_rdarea_list_no_wrn_pnt

(15) Operands related to lock

Operand explanation	Operand name
Specifies whether or not deadlock information is to be output.	pd_lck_deadlock_info
Specifies the maximum amount of time for monitoring lock wait time.	pd_lck_wait_timeout
Specifies the method by which HiRDB is to detect lock release.	pd_lck_release_detect
Specifies the interval at which HiRDB is to reference the lock management area.	pd_lck_release_detect_interval
Specifies the processing method to be used when WITHOUT LOCK NOWAIT search is executed.	pd_nowait_scan_option
Specifies the number of users who must be waiting for lock release in order for a warning message to be issued.	pd_lck_queue_limit
Specifies whether or not deadlock priorities are to be used.	pd_deadlock_priority_use

Operand explanation	Operand name
Specifies the deadlock priority value of a command.	pd_command_deadlock_priority
Specifies the method for creating a locked resource for an index key value.	pd_key_resource_type
Specifies the amount of lock pool space to be used by a server.	pd_lck_pool_size
Specifies the amount of lock pool to be used by a front-end server.	pd_fes_lck_pool_size
Specifies the number of tables and RDAREAs to be held across transactions and the number of resources to be locked for inner replica configuration management and replica group configuration management.	pd_lck_until_disconnect_cnt
Specifies the maximum number of holdable cursors that can be concurrently open for each transaction when holdable cursors are used for a table for which a LOCK statement with UNTIL DISCONNECT specification is not executed.	pd_max_open_holdable_cursors
Specifies the number of hash table entries to be used in the lock pool.	pd_lck_hash_entry
Specifies the locking method for a B-tree index (whether or not "index key value no lock" is to be executed).	pd_indexlock_mode
Specifies an interval for unlocking global buffers when global buffer locking occurs during synchronization point processing.	pd_dbsync_lck_release_count

(16) Operands related to buffers

Operand explanation	Operand name
Specifies the size of the buffer area in which SQL objects are to be placed.	pd_sql_object_cache_size
Specifies the size of the buffer for table definition information.	pd_table_def_cache_size
Specifies whether the control areas for the buffers for table definition information, buffers for view analysis information, buffers for user-defined type information, and buffers for routine definition information are to be allocated in a single batch at the time of HiRDB activation or in a single batch at the time of transaction activation.	pd_def_buf_control_area_assign
Specifies the maximum stack size to be used by one thread.	pd_thred_max_stack_size
Specifies the size of the buffer for user privilege information.	pd_auth_cache_size
Specifies the size of the buffer for view analysis information.	pd_view_def_cache_size
Specifies the size of the buffer for table alias definition information.	pd_alias_cache_size
Specifies the size of the buffer for user-defined type information.	pd_type_def_cache_size

Operand explanation	Operand name
Specifies the size of the buffer for routine definition information.	`pd_routine_def_cache_size`
Specifies the size of a buffer to be used to store registry information.	`pd_registry_cache_size`

(17) Operands related to shared memory

Operand explanation	Operand name
Specifies whether or not the shared memory to be used by the HiRDB is to be fixed in the memory.	`pd_shmpool_attribute`
Specifies whether or not the shared memory to be used by the global buffer is to be fixed in the real memory.	`pd_dbbuff_attribute`
Specifies the size of the shared memory to be used by a single server.	`pd_sds_shmpool_size`
Specifies the size of the shared memory to be used by the dictionary server.	`pd_dic_shmpool_size`
Specifies the size of the shared memory to be used by a back-end server.	`pd_bes_shmpool_size`
Specifies the upper limit of the segment size for the shared memory for the global buffer pool.	`SHMMAX`

(18) Operands related to RDAREAs

Operand explanation	Operand name
Specifies the maximum number of RDAREAs allowed.	`pd_max_rdarea_no`
Specifies the maximum number of HiRDB files that comprise an RDAREA.	`pd_max_file_no`
Specifies the trigger for issuing a warning message (`KFPH00211-I` or `KFPA12300-I`) about the segment usage ratio.	`pd_rdarea_warning_point`
Specifies whether to use the `DEFER` or `SCHEDULE` attribute as the RDAREA opening trigger.	`pd_rdarea_open_attribute_use`
Specifies the standard value for the RDAREA opening trigger attribute.	`pd_rdarea_open_attribute`
Specifies whether to use a shared RDAREA.	`pd_shared_rdarea_use`

(19) Operands related to global buffers

Operand explanation	Operand name
Specifies the LRU management method for the global buffer.	`pd_dbbuff_lru_option`

Operand explanation	Operand name
Specifies whether to dynamically modify the global buffer.	`pd_dbbuff_modify`
Specifies the method of detecting lock release of the global buffer.	`pd_dbbuff_lock_release_detect`
Specifies the number of spins during lock acquisition wait processing for the global buffer.	`pd_dbbuff_lock_spn_count`
Specifies the interval during lock acquisition wait processing for the global buffer.	`pd_dbbuff_lock_interval`
Specifies the interval at which to check the global buffer occupation state.	`pd_dbbuff_wait_interval`
Specifies the maximum number of spin loops for global buffer occupation state checking.	`pd_dbbuff_wait_spn_count`
Specifies the deferred write trigger event as a percentage of the updated buffers.	`pd_dbbuff_rate_updpage`
Defines the RDAREAs to be allocated to a global buffer.	`pdbuffer`
Specifies the maximum number of global buffers for dynamic addition.	`pd_max_add_dbbuff_no`
Specifies the maximum number of shared memory segments for dynamic addition.	`pd_max_add_dbbuff_shm_no`

(20) Operands related to table or index reservation count

Operand explanation	Operand name
Specifies the minimum guaranteed value for the table reservation count.	`pd_assurance_table_no`
Specifies the guaranteed minimum number of indexes.	`pd_assurance_index_no`

(21) Operands related to referential and check constraints

Operand explanation	Operand name
Specifies whether to specify a constraint name definition before or after constraint definition in a referential constraint or check constraint.	`pd_constraint_name`
Specified when a referential constraint or check constraint is used.	`pd_check_pending`

(22) Operands related to HiRDB file system areas

Operand explanation	Operand name
Specifies whether or not large files can be used.	pd_large_file_use

(23) Operands related to system log files

Operand explanation	Operand name
Specifies whether or not dual system log files are to be used.	pd_log_dual
Specifies whether to use the facility for parallel output of system logs.	pd_log_dual_write_method
Specifies the processing to be performed by HiRDB when the available space in the system log file falls below the warning level (the facility for monitoring the free area for the system log file).	pd_log_remain_space_check
Specifies as absolute pathnames the unload file output directories or HiRDB file system areas when the automatic log unloading facility is to be used for the system log.	pd_log_auto_unload_path
Specifies whether or not single operation of the system log files is to be used.	pd_log_singleoperation
Specifies whether or not a reserved file is to be used when none of the opened files can be used for swapping.	pd_log_rerun_reserved_file_open
Specifies whether or not system log files are to be swapped during a restart.	pd_log_rerun_swap
Specifies a wait time for swapping of system log files to be completed.	pd_log_swap_timeout
Specifies whether or not the unload status of system log files is to be checked. If unload status checking is not performed, system log information is not available for database recovery.	pd_log_unload_check
Specifies the size of the buffer to be used for system log input/output.	pd_log_max_data_size
Specifies the number of buffer sectors to be used for system log output.	pd_log_write_buff_count
Specifies the system log file record length.	pd_log_rec_leng
Specifies the name of a file group for system log files.	pdlogadfg -d sys
Specifies the name of the system log file to be allocated to a file group.	pdlogadpf -d sys

(24) Operands related to synchronization point dump files

Operand explanation	Operand name
Specifies whether to use dual synchronization point dump files.	pd_spd_dual

Operand explanation	Operand name
Specifies whether or not a message is to be output when a synchronization point dump is completed.	`pd_spd_assurance_msg`
Specifies the number of guaranteed valid generations.	`pd_spd_assurance_count`
Specifies whether or not the reduced mode of operation is to be used.	`pd_spd_reduced_mode`
Specifies whether or not a reserved file is to be opened automatically.	`pd_spd_reserved_file_auto_open`
Specifies the size of the buffer to be used for synchronization point dump input/output.	`pd_spd_max_data_size`
Specifies the synchronization point dump collection interval.	`pd_log_sdinterval`
Specifies the name of a file group for a synchronization point dump file.	`pdlogadfg -d spd`
Specifies the name of a synchronization point dump file to be allocated to a file group.	`pdlogadpf -d spd`
Specifies a group name for synchronization point dump files for log application processing.	`pdlogadfg -d ssp`
Specifies a synchronization point dump file for log application processing that is to be allocated to a file group.	`pdlogadpf -d ssp`

(25) Operands related to status files

Operand explanation		Operand name
Unit status files	Specify the names of unit status files.	`pd_syssts_file_name_1-7`
	Specify the names of unit duplicate status file.	`pd_syssts_subfile_name_1-7`
	Specifies the action to be taken when an error is detected in a unit status file during unit startup.	`pd_syssts_initial_error`
	Specifies whether or not single operation of unit status files is to be used.	`pd_syssts_singleoperation`
	Specifies the name of the unit status file to be used as the current file.	`pd_syssts_last_active_file`
	During a restart, specifies the unit status file that was normal during the previous operation.	`pd_syssts_last_active_side`
	Specifies the name of the unit duplicate status file to be used as the current file.	`pd_syssts_last_active_subfile`

Operand explanation		Operand name
	During a restart, specifies the unit duplicate status file that was normal during the previous operation.	pd_syssts_last_active_side_sub
Server status files	Specify the names of server status files.	pd_sts_file_name_1-7
	Specify the names of server duplicate status file.	pd_sts_subfile_name_1-7
	Specifies the action to be taken when an error is detected in a server status file during server startup.	pd_sts_initial_error
	Specifies whether or not single operation of server status files is to be used.	pd_sts_singleoperation
	Specifies the name of the server status file to be used as the current file.	pd_sts_last_active_file
	During a restart, specifies the server status file that was normal during the previous operation.	pd_sts_last_active_side
	Specifies the name of the server duplicate status file to be used as the current file.	pd_sts_last_active_subfile
	During a restart, specifies the server duplicate status file that was normal during the previous operation.	pd_sts_last_active_side_sub

(26) Operand related to message log file

Operand explanation	Operand name
Specifies a message log output destination unit.	pd_mlg_msg_log_unit
Specifies the maximum size of the message log file.	pd_mlg_file_size

(27) Operands related to statistical information

Operand explanation	Operand name
Specifies whether or not collection of statistical information is to begin at the time of HiRDB startup.	pd_statistics
Specifies the maximum size of the statistics log file.	pd_stj_file_size
Specifies the statistics log buffer size.	pd_stj_buff_size
Specifies the statistical information that is to be output, beginning at the time of HiRDB startup.	pdstbegin
Specifies that collection of statistical information related to CONNECT and DISCONNECT is to begin at the time of HiRDB startup.	pdhibegin

(28) Operands related to RPC trace information

Operand explanation	Operand name
Specifies whether or not RPC trace information is to be collected.	pd_rpc_trace
Specifies the name of the file in which RPC trace information is to be collected.	pd_rpc_trace_name
Specifies the size of the file in which RPC trace information is to be collected.	pd_rpc_trace_size

(29) Operands related to troubleshooting information

Operand explanation	Operand name
Specifies whether to output troubleshooting information.	pd_cancel_dump
Specifies whether to collect the following types of troubleshooting information when the client maximum wait time (value of the PDCWAITTIME operand in the client environment definition) is exceeded during transaction execution.	pd_client_waittime_over_abort
Specifies the amount of time during which to suppress re-outputting the troubleshooting information.	pd_dump_suppress_watch_time
Specifies whether or not the network information portion of troubleshooting information is to be collected when a HiRDB process or HiRDB (unit) terminates abnormally.	pd_debug_info_netstat
Specifies the interval at which to delete the troubleshooting information files (files under $PDDIR/spool) and the temporary work files (files under $PDDIR/tmp) that are output by HiRDB.	pd_spool_cleanup_interval
Specifies the files to be deleted by the pd_spool_cleanup_interval operand.	pd_spool_cleanup_interval_level
Specifies whether to delete during the startup of HiRDB the troubleshooting information files (files under $PDDIR/spool) that are output by HiRDB.	pd_spool_cleanup
Specifies the troubleshooting information files to be deleted by the pd_spool_cleanup operand.	pd_spool_cleanup_level
Specifies the maximum number of module trace records to be stored.	pd_module_trace_max
Specifies how to acquire the time to be output in module traces.	pd_module_trace_timer_level

(30) Operands related to the BES connection holding facility

Operand explanation	Operand name
Specifies whether to use the BES connection holding facility.	pd_bes_connection_hold

Operand explanation	Operand name
Specifies the BES connection holding period.	`pd_bes_conn_hold_trn_interval`

(31) Operand related to the facility for predicting reorganization time

Operand explanation	Operand name
Specifies whether to use the facility for predicting reorganization time.	`pd_rorg_predict`

(32) Operands related to security audit facility

Operand explanation	Operand name
Specifies whether to begin collecting an audit trail when HiRDB (a unit for a HiRDB/Parallel Server) is started.	`pd_audit`
Specifies an absolute path name for the name of the HiRDB file system area for an audit trail file.	`pd_aud_file_name`
Specifies the maximum size (units: MB) for audit trail files.	`pd_aud_max_generation_size`
Specifies the maximum number of (number of generations of) audit trail files to be created inside the HiRDB file system area for audit trail files.	`pd_aud_max_generation_num`
Specifies the process to be performed by HiRDB when no swappable audit trail file is available.	`pd_aud_no_standby_file_opr`
Specifies the size (units: bytes) of the buffer to be used for asynchronously outputting an audit trail.	`pd_aud_async_buff_size`
Specifies the number of buffer sectors to be used for asynchronously outputting an audit trail.	`pd_aud_async_buff_count`
Specifies the retry interval for monitoring for a buffer to be used for asynchronous output of an audit trail in order to acquire an audit trail when all buffers are in use.	`pd_aud_async_buff_retry_intvl`
When the number of unswappable audit trail files reaches or exceeds the warning value, a warning message is issued. For this operand, specify the warning value as a percentage of the maximum audit trail file count specified in the `pd_aud_max_generation_num` operand.	`pd_aud_file_wrn_pnt`
Specifies the buffer size to be used to store information for the security audit facility.	`pd_audit_def_buffer_size`

(33) Operands related to system switchover facility

Operand explanation	Operand name
Specifies whether or not the system switchover facility is to be used.	`pd_ha`

Operand explanation	Operand name
Specifies whether or not IP addresses are to be inherited when the system switchover facility is used.	pd_ha_ipaddr_inherit
Specifies that the system switchover facility is not to be applied to the unit.	pd_ha_unit
Specifies the absolute path name of the command to be executed when HiRDB restart fails.	pd_ha_restart_failure
Specifies whether to run the system switchover facility in the monitor mode or server mode.	pd_ha_acttype
Specifies whether to use user server hot standby.	pd_ha_server_process_standby
Specifies whether to use the rapid system switchover facility, standby-less system switchover (1:1) facility, or standby-less system switchover (effects distributed) facility.	pd_ha_agent
Specifies the maximum number of guest BESs that operate as running systems inside the unit when the standby-less system switchover (effects distributed) facility is used.	pd_ha_max_act_guest_servers
Specifies the maximum number of user server processes to be started inside a unit when the standby-less system switchover (effects distributed) facility is used.	pd_ha_max_server_process
Specifies the combined total of the number of host BESs inside the unit and the number of resident processes for the guest BESs after these guest BESs have been accepted when the standby-less system switchover (effects distributed) facility is used.	pd_ha_process_count
When the standby-less system switchover (effects distributed) facility is used, this operand specifies a maximum amount of time to wait until the running server's resources are activated at the time of unit startup.	pd_ha_resource_act_wait_time
Specifies whether to use the transaction queuing facility. Also specifies the processing that takes place when the number of connections to the HiRDB server exceeds the maximum number of concurrent connections (value specified by the pd_max_users operand) during system switchover.	pd_ha_transaction
Specifies the transaction queuing wait time when the transaction queuing facility is used.	pd_ha_trn_queuing_wait_time
Specifies the upper limit for retrying transaction start requests after system switchover occurs and before the standby unit restarts.	pd_ha_trn_restart_retry_time
Specifies whether to switch the system without waiting for internal termination processing of HiRDB when internal termination processing of HiRDB (or a unit for a HiRDB/Parallel Server) during system switchover has exceeded the server failure monitoring time.	pd_ha_switch_timeout

Operand explanation	Operand name
When the server-mode system switchover facility is used, this operand specifies whether or not system switchover processing is to be placed on hold until all server processes subject to switchover have terminated.	`pd_ha_prc_cleanup_check`
Specifies whether to wait for the completion of start processing of other units before switching the system for the system manager unit.	`pd_ha_mgr_rerun`
Defines an HA group.	`pdhagroup`
Specifies that the system switchover facility is not to be applied to the unit.	`pdunit`
Specifies the server machine to be used for executing the servers that comprise a HiRDB system and how each server machine is to be used.	`pdstart`
Specifies the name of the standard host of the primary system.	`pd_hostname`

(34) Operands related to HiRDB Datareplicator

Operand explanation	Operand name
Specifies whether or not the HiRDB Datareplicator linkage facility is to used from the time of HiRDB startup.	`pd_rpl_init_start`
When the HiRDB Datareplicator linkage facility is used, this operand specifies the unit for applying transactions.	`pd_rpl_reflect_mode`
Specifies the action to be taken when a swap request is received when not all of the system log files can be created at the swapping destination because system log extraction by HiRDB Datareplicator using the HiRDB Datareplicator linkage facility has not been completed.	`pd_log_rpl_no_standby_file_opr`
Specifies the extracted side HiRDB Datareplicator directory when the HiRDB Datareplicator linkage facility is used.	`pd_rpl_hdepath`

(35) Operands related to linkage to JP1

Operand explanation	Operand name
Specifies whether or not HiRDB events are to be registered into JP1.	`pd_jp1_use`
Specifies the HiRDB event types that are to be registered into JP1.	`pd_jp1_event_level`
Specifies whether to register message-outputting events into JP1.	`pd_jp1_event_msg_out`

(36) Operands related to Directory Server linkage facility

Operand explanation	Operand name
Specifies that the Directory Server linkage facility is to be used.	pd_directory_server

(37) Operands related to Real Time SAN Replication

Operand explanation	Operand name
Specifies whether to use Real Time SAN Replication.	pd_rise_use
Selects a Real Time SAN Replication processing method.	pd_rise_pairvolume_combination
Specifies whether to maintain data matching by establishing synchronization between the main site and the remote site when the hybrid method is selected for the Real Time SAN Replication processing method.	pd_rise_disaster_mode
Specifies the action to be taken when an attempt to update a volume located at a remote site by synchronously copying data fails because of an error when the synchronous method or the hybrid method is selected for the Real Time SAN Replication processing method.	pd_rise_fence_level
Specifies the instance number of the RAID Manager that defines the pair logical volume to be used by HiRDB.	HORCMINST

(38) Operands related to inner replica facility

Operand explanation	Operand name
Specifies the maximum number of inner replica groups.	pd_inner_replica_control
Specifies whether to use the mirror facility of Logical Volume Manager (LVM) to run the replica RDAREA.	pd_lv_mirror_use
Specifies the number of processes to be used for reflection processing.	pd_max_reflect_process_count
Specifies whether to change the status of the system log file for which reflection processing has been completed.	pd_log_org_reflected_logpoint
Specifies the processing for HiRDB when system log file swapping occurs while all system log files are in the overwriting denied status for online reorganization.	pd_log_org_no_standby_file_opr

(39) Operands related to HiRDB External Data Access facility

Operand explanation	Operand name
Specifies the maximum number of foreign servers that can be connected.	pd_max_foreign_server

Operand explanation	Operand name
Specifies the path name for a foreign server client library.	pd_foreign_server_libpath
Specifies the foreign server to which the Hub optimization information definition is applied.	pdhubopt
Defines the execution environment for the HiRDB External Data Access facility.	Foreign server information definition
Defines whether to use a foreign server to execute each syntax of the input query. Also defines the values of the parameters necessary for cost expression computation.	Hub optimization information definition

(40) Operand related to OLTP

Operand explanation	Operand name
Specifies whether to use the holdable cursor facility in a UAP under the OLTP environment.	pd_oltp_holdcr

(41) Operand related to distributed database

Operand explanation	Operand name
Specifies the local node's RD node name.	pd_node_name

(42) Operand related to date and time

Operand explanation	Operand name
Specifies the time zone.	TZ

(43) Operand related to the message output suppression facility

Operand explanation	Operand name
Controls whether to allow HiRDB to output messages in syslogfile.	pdmlgput

(44) Operand related to HiRDB file system sharing

Operand explanation	Operand name
Specifies the name of the directory that is to be shared when sharing of HiRDB system definition files is in effect.	PDCONFPATH

(45) Operand to client group

Operand explanation	Operand name
Specifies the number of users who are guaranteed connection when the connection frame guarantee facility for a client group is used.	pdcltgrp

(46) Operands related to plug-in

Operand explanation	Operand name
Specifies the name of a plug-in to be used.	pdplugin
Specifies the size of the shared memory to be used by a plug-in.	pdplgprm
Specifies the directory in which the index information file is to be created when delayed batch creation of plug-in indexes is to be performed.	pd_plugin_ixmk_dir

(47) Operands related to version upgrade

Operand explanation	Operand name
Specifies whether or not the pdvrup command for executing HiRDB version upgrading is to be started automatically.	pd_auto_vrup
Specifies whether to keep the default values of the system definition operands of HiRDB Version 6 or earlier.	pd_sysdef_default_option

(48) Operands related to communication processing

Operand explanation	Operand name
Specifies a port number unique within the host when using a client that uses the high-speed connection facility.	pd_service_port
Specifies the network to be used by the HiRDB server to communicate with HiRDB clients.	pd_change_clt_ipaddr
Specifies the range of port numbers to be used for communication by the HiRDB server.	pd_registered_port
Specifies whether or not checking is to be performed for overlapping port numbers in the ranges of port numbers specified in the pd_registered_port operand and in the port numbers registered in /etc/services.	pd_registered_port_check
Specifies a target range for the HiRDB reserved port facility.	pd_registered_port_level
Specifies the number of times process-to-process communication can be attempted.	pd_ipc_send_retrycount

Operand explanation	Operand name
Specifies the sleep time between process-to-process send retries.	pd_ipc_send_retrysleeptime
Specifies the number of times server-to-server send can be performed before send is terminated.	pd_ipc_send_count
Specifies the number of times server-to-server receive can be performed before receive is terminated.	pd_ipc_recv_count
Specifies whether to establish connection in the non-block mode for HiRDB server-to-server communication (communication between units).	pd_ipc_conn_nblock
Specifies whether or not connection is to be established in the non-block mode for HiRDB server-to-server communication (communication between units).	pd_ipc_conn_nblock_time
Specifies the retry interval for establishing connection for sending data.	pd_ipc_conn_interval
Specifies the number of retries for establishing connection for sending data.	pd_ipc_conn_count
Specifies the maximum size for the send/receive buffer to be used for unit-to-unit communication within a server (TCP INET domain).	pd_ipc_inet_bufsize
Specifies the maximum size for the send/receive buffer to be used for communication within a server unit (TCP UNIX domain).	pd_ipc_unix_bufsize
Specifies the maximum size for the send/receive buffer to be used for communication with a HiRDB client connected from outside the host where the HiRDB server resides (TCP INET domain).	pd_tcp_inet_bufsize
Specifies the maximum size for the send/receive buffer to be used for communication with a HiRDB client of HiRDB Version 4.0 04-02 or earlier connected from the host where the HiRDB server resides (TCP UNIX domain).	pd_tcp_unix_bufsize
Specifies the size of the communication buffer used by utility processes.	pd_utl_buff_size
Specifies the size of the communication buffer used for transferring retrieved data among HiRDB servers.	pd_sql_send_buff_size
For the AIX 5L version, specifies whether or not immediate acknowledgments are to be sent.	pd_ipc_tcp_nodelayack

(49) Operands related to Java

Operand explanation	Operand name
Specifies Java virtual option startup options.	pd_java_option

Operand explanation	Operand name
Specifies in bytes the stack size to be used by a Java routine.	`pd_java_routine_stack_size`
Specifies the name of the directory for storing JAR files used by Java stored procedures or Java stored functions.	`pd_java_archive_directory`
Specifies the class path to be used by a Java virtual machine.	`pd_java_classpath`
Specifies the root directory of the Java Runtime Environment as an absolute pathname.	`pd_java_runtimepath`
Specifies the directory that stores the library of the Java virtual machine as a relative pathname to the Java Runtime Environment root directory.	`pd_java_libpath`
Specifies the file to which the standard output and standard error output are to be output in a Java virtual machine.	`pd_java_stdout_file`

(50) Operands related to SQL runtime warning output facility

Operand explanation	Operand name
Specifies the condition for outputting SQL runtime warning information.	`pd_cwaittime_wrn_pnt`
Specifies the output destination directory for the SQL runtime warning information file.	`pd_cwaittime_report_dir`
Specifies the maximum size for the SQL runtime warning information file.	`pd_cwaittime_report_size`

(51) Operands related to local buffers

Operand explanation	Operand name
Specifies the action to be taken by the UAP when the RDAREA or index to be accessed using a local buffer is being used by another user.	`pd_uap_wait`
Specifies the local buffer used by a UAP.	`pdlbuffer`

(52) Operands related to character encoding

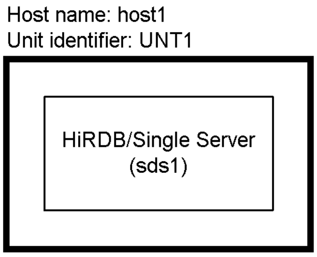
Operand explanation	Operand name
Specifies the maximum number of bytes to be used to represent a single character.	`pd_substr_length`

C. Examples of Definitions

C.1 Example 1 (HiRDB/Single Server)

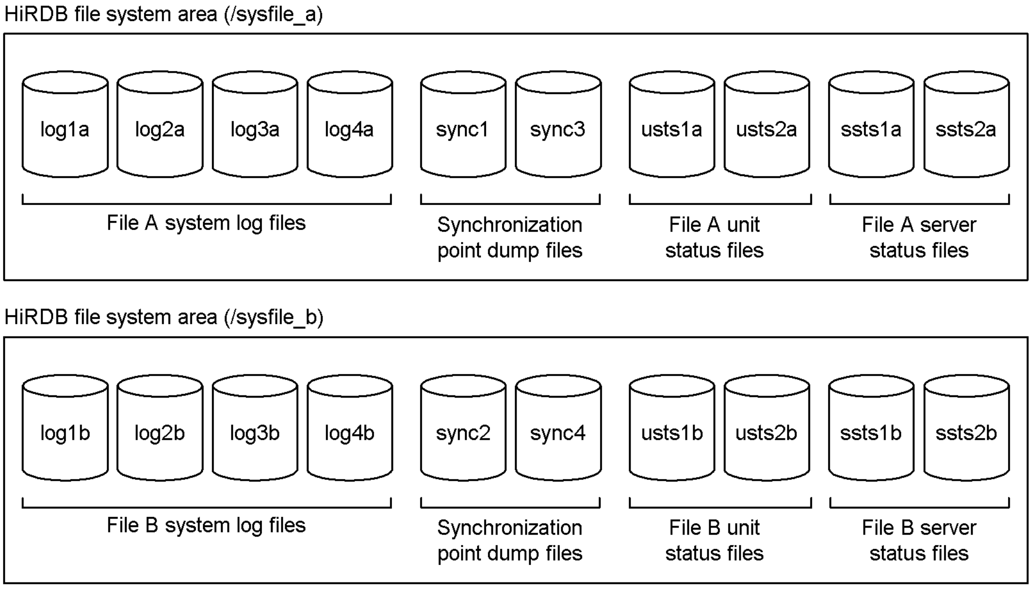
This example creates system definitions for a HiRDB/Single Server. The system configuration is shown as follows:

Unit configuration



Note: Server name is shown in parentheses.

System files configuration



(1) System common definition

```
set pd_system_id = PDB1                                         1
set pd_name_port = 20001                                        2
set pd_master_file_name = "/rdarea/mast/mast01"                 3
set pd_max_users = 100                                          4
set pd_max_server_process = 220                                 5
set pd_max_access_tables = 50                                   6
set pd_sql_object_cache_size = 3000                             7
set pd_max_rdarea_no = 200                                      8
set pd_max_file_no = 600                                        9
set pd_optimize_level = "PRIOR_NEST_JOIN", \                    10
    "PRIOR_OR_INDEXES","DETER_AND_INDEXES", \
    "RAPID_GROUPING","DETER_WORK_TABLE_FOR_UPDATE", \
    "APPLY_ENHANCED_KEY_COND"
set pd_additional_optimize_level = "COST_BASE_2"                11
pdunit -x host1 -u UNT1 -d "/HiRDB"                             12
pdstart -t SDS -s sds1 -u UNT1                                  13
pdbuffer -a gbuf01 -r RDMAST,RDDIC,RDDIR -n 1000                14
pdbuffer -a gbuf02 -r RDAREA1,RDAREA2,RDAREA3 -n 1000           15
pdbuffer -a gbuf03 -r RDAREA4,RDAREA5,RDAREA6 -n 1000           16
pdbuffer -a gbuf04 -o -n 1000                                   17
putenv SHMMAX 16                                                18
```

Explanation

1. Specifies the HiRDB identifier.
2. Specifies the HiRDB port number.
3. Specifies the first HiRDB file in the master directory RDAREA.
4. Specifies the maximum number of concurrent connections.
5. Specifies the maximum number of server processes that can be activated concurrently.
6. Specifies the maximum number of base tables that can be accessed concurrently.
7. Specifies the SQL object buffer size.
8. Specifies the maximum number of RDAREAs allowed.
9. Specifies the maximum number of HiRDB files that comprise an RDAREA.
10. Specifies SQL optimization options.
11. Specifies an SQL extension optimizing option.
12. Specifies the unit configuration:

    `-x`: Host name

    `-u`: Unit identifier

`-d`: HiRDB directory name

13. Specifies the server configuration:

    `-t`: Server type (SDS: single server)

    `-s`: Server name

    `-u`: Unit identifier

14. Allocates the master directory RDAREA, data dictionary RDAREA, and data directory RDAREA to a global buffer.
15. Allocates user RDAREAs (RDAREA1-RDAREA3) to a global buffer.
16. Allocates user RDAREAs (RDAREA4-RDAREA6) to a global buffer.
17. Allocates other RDAREAs to a global buffer.
18. Specifies the maximum number of shared memory segments.

(2) Unit control information definition

```
set pd_unit_id = UNT1                                                1
set pd_syssts_file_name_1 = "untsts1","/sysfile_a/usts1a",\          2
                                       "/sysfile_b/usts1b"
set pd_syssts_file_name_2 = "untsts2","/sysfile_a/usts2a",\
                                       "/sysfile_b/usts2b"
```

Explanation

1. Specifies the unit identifier.
2. Specifies the configuration of the unit status files.

(3) Single server definition

```
set pd_sds_shmpool_size = 10000                                        1
set pd_log_dual = Y                                                    2
set pd_sts_file_name_1 = "svrsts1","/sysfile_a/ssts1a",\               3
                                   "/sysfile_b/ssts1b"
set pd_sts_file_name_2 = "svrsts2","/sysfile_a/ssts2a",\
                                   "/sysfile_b/ssts2b"
pdwork -v "/work01","/work02"                                          4
pdlogadfg -d sys -g log1 ONL                                           5
pdlogadfg -d sys -g log2 ONL
pdlogadfg -d sys -g log3 ONL
pdlogadfg -d sys -g log4 ONL
pdlogadpf -d sys -g log1 -a "/sysfile_a/log1a" -b "/sysfile_b/log1b"
pdlogadpf -d sys -g log2 -a "/sysfile_a/log2a" -b "/sysfile_b/log2b"
pdlogadpf -d sys -g log3 -a "/sysfile_a/log3a" -b "/sysfile_b/log3b"
pdlogadpf -d sys -g log4 -a "/sysfile_a/log4a" -b "/sysfile_b/log4b"
pdlogadfg -d spd -g sync1 ONL                                          6
pdlogadfg -d spd -g sync2 ONL
pdlogadfg -d spd -g sync3 ONL
pdlogadfg -d spd -g sync4 ONL
pdlogadpf -d spd -g sync1 -a "/sysfile_a/sync1"
pdlogadpf -d spd -g sync2 -a "/sysfile_b/sync2"
pdlogadpf -d spd -g sync3 -a "/sysfile_a/sync3"
pdlogadpf -d spd -g sync4 -a "/sysfile_b/sync4"
```

Explanation

1. Specifies the size of the shared memory to be used by the single server.
2. Specifies use of dual system log files.
3. Specifies the configuration of the server status files.
4. Specifies HiRDB file system areas for work table files.
5. Specifies the system log files configuration.
6. Specifies the synchronization point dump files configuration.

(4) UAP environment definition

```
set pd_uap_wait = Y                                        1
pdlbuffer -a localbuf1 -r RDAREA10 -n 1000 -p 16           2
pdlbuffer -a localbuf2 -r RDAREA11,RDAREA12 -n 1000        3
pdlbuffer -a localbuf3 -i USER01.INDX01 -n 1000            4
```

Explanation

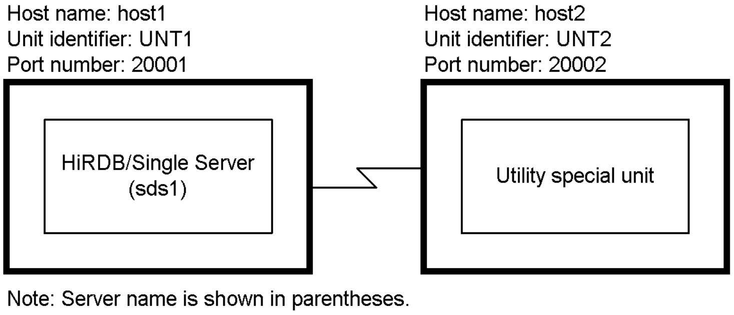
1. Specifies the action to be taken by the UAP when the RDAREA or index to be accessed using a local buffer is being used by another user.

2. Allocates a local buffer to a user RDAREA (RDAREA10).
3. Allocates a local buffer to user RDAREAs (RDAREA11 and RDAREA12).
4. Allocates a local buffer to an index (INDX01).

C.2 Example 2 (HiRDB/Single Server: with utility special unit)

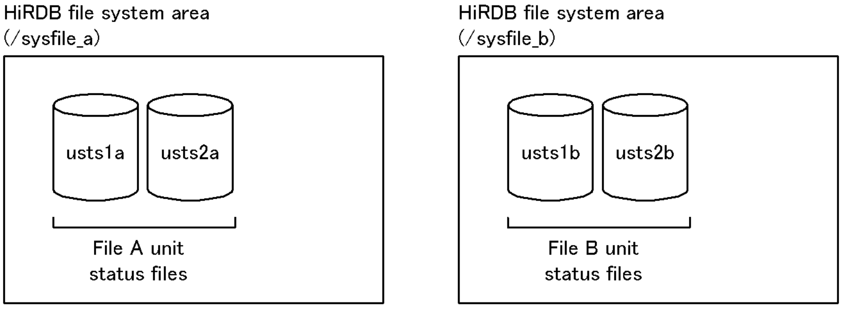
In this example, a utility special unit is installed in the HiRDB/Single Server defined in Example 1. System definitions for the HiRDB/Single Server and the utility special unit are created. The system configuration is shown below:

Unit configuration



Note: Server name is shown in parentheses.

System files configuration for the utility special unit



(1) System common definition for the HiRDB/Single Server

```
             :
pdunit -x host2 -u UNT2 -d "/HiRDB" -p 20002
             :
```

Explanation

This coding modifies the system common definition created in Example 1 (by adding the pdunit operand).

The added pdunit operand defines the unit configuration of the utility special unit.

(2) System common definition for the utility special unit

```
set pd_system_id = PDB2                                                1
set pd_name_port = 20002                                               2
pdunit -x host1 -u UNT1 -d "/HiRDB" -p 20001                           3
pdunit -x host2 -u UNT2 -d "/HiRDB"                                    4
pdstart -t SDS -s sds1 -u UNT1                                         5
```

Explanation

1. Specifies the HiRDB identifier.
2. Specifies the HiRDB port number.
3. Specifies the unit configuration in the HiRDB/Single Server.
4. Specifies the unit configuration in the utility special unit.
5. Specifies the server configuration in the HiRDB/Single Server.

(3) Unit control information definition for the utility special unit

```
set pd_unit_id = UNT2                                                  1
set pd_syssts_file_name_1 = "untsts1","/sysfile_a/usts1a",\            2
                                      "/sysfile_b/usts1b"
set pd_syssts_file_name_2 = "untsts2","/sysfile_a/usts2a",\
                                      "/sysfile_b/usts2b"
```

Explanation

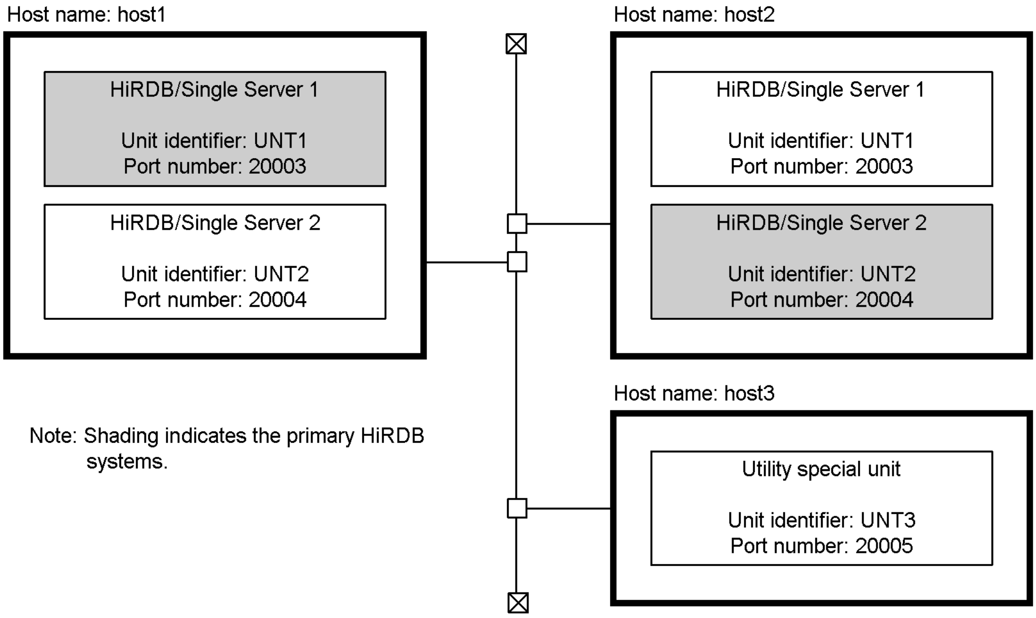
1. Specifies the unit identifier.
2. Specifies the configuration of the unit status files.

C.3 Example 3 (HiRDB/Single Server: system switchover facility being used)

This example creates system definitions for a HiRDB/Single Server. The system configuration is shown as follows:

- This configuration is based on mutual system switchover between HiRDB/Single Server 1 and HiRDB/Single Server 2.
- Utility special units used by HiRDB/Single Server 1 and HiRDB/Single Server 2 are installed.
- The system switchover facility does not inherit IP addresses.

This example explains only the system common definition, unit control definition, and their related operands.



(1) System common definition of HiRDB/Single Server 1

```
set pd_system_id = SDS1                                         1
set pd_name_port = 20003                                        2
         :
         :
set pd_ha = use                                                 3
set pd_ha_ipaddr_inherit = N                                    4
         :
         :
pdunit -x host1 -u UNT1 -d "/HiRDB1" -c host2                   5
pdunit -x host3 -u UNT3 -d "/HiRDB3" -p 20005                   6
pdstart -t SDS -s sds01 -u UNT1                                 7
```

Explanation

1. Specifies the HiRDB identifier.
2. Specifies the HiRDB port number.
3. Specifies use of the system switchover facility.
4. Specifies that IP addresses are not inherited.
5. Specifies the unit configuration of the HiRDB/Single Server 1:

-x: Host name.

-u: Unit identifier

-d: HiRDB directory name

-c: Host name of the secondary system

6. Specifies the configuration of the utility special unit:

   -p: Port number

7. Specifies the server configuration of the HiRDB/Single Server 1:

   -t: Server type

   -s: Server name

   -u: Unit identifier

(2) Unit control information definition of HiRDB/Single Server 1

```
set pd_unit_id = UNT1                                      1
         :
         :
set pd_hostname = host1                                    2
```

Explanation

1. Specifies the unit identifier.
2. Specifies the host name of the primary system.

(3) System common definition of HiRDB/Single Server 2

```
set pd_system_id = SDS2                                    1
set pd_name_port = 20004                                   2
         :
         :
set pd_ha = use                                            3
set pd_ha_ipaddr_inherit = N                               4
         :
         :
pdunit -x host2 -u UNT2 -d "/HiRDB2" -c host1              5
pdunit -x host3 -u UNT3 -d "/HiRDB3" -p 20005              6
pdstart -t SDS -s sds02 -u UNT2                            7
```

Explanation

1. Specifies the HiRDB identifier.
2. Specifies the HiRDB port number.

3. Specifies use of the system switchover facility.
4. Specifies that IP addresses are not inherited.
5. Specifies the unit configuration of the HiRDB/Single Server 2:

   `-x`: Host name.

   `-u`: Unit identifier

   `-d`: HiRDB directory name

   `-c`: Host name of the secondary system
6. Specifies the configuration of the utility special unit:

   `-p`: Port number
7. Specifies the server configuration of the HiRDB/Single Server 2:

   `-t`: Server type

   `-s`: Server name

   `-u`: Unit identifier

(4) Unit control information definition of HiRDB/Single Server 2

```
set pd_unit_id = UNT2                                        1
        :
        :
set pd_hostname = host2                                      2
```

Explanation

1. Specifies the unit identifier.
2. Specifies the host name of the primary system.

(5) System common definitions of the utility special units

```
set pd_system_id = UTL1                                     1
set pd_name_port = 20005                                    2
         :
         :
set pd_ha = use                                             3
set pd_ha_ipaddr_inherit = N                                4
         :
         :
pdunit -x host1 -u UNT1 -d "/HiRDB1" -c host2 -p 20003      5
pdunit -x host2 -u UNT2 -d "/HiRDB2" -c host1 -p 20004      6
pdunit -x host3 -u UNT3 -d "/HiRDB3"                        7
pdstart -t SDS -s sds01 -u UNT1                             8
pdstart -t SDS -s sds02 -u UNT2                             9
```

Explanation

1. Specifies the HiRDB identifier.
2. Specifies the HiRDB port number.
3. Specifies use of the system switchover facility.
4. Specifies that IP addresses are not inherited.
5. Specifies the unit configuration of the HiRDB/Single Server 1:

   `-x`: Host name.

   `-u`: Unit identifier

   `-d`: HiRDB directory name

   `-c`: Host name of the secondary system

   `-p`: Port number
6. Specifies the unit configuration of the HiRDB/Single Server 2:
7. Specifies the configuration of the utility special unit:
8. Specifies the server configuration of the HiRDB/Single Server 1:

   `-t`: Server type

   `-s`: Server name

   `-u`: Unit identifier
9. Specifies the server configuration of the HiRDB/Single Server 2:

(6) Unit control information definitions of the utility special units

```
set pd_unit_id = UNT3                                        1
        :
        :
set pd_hostname = host3                                      2
```

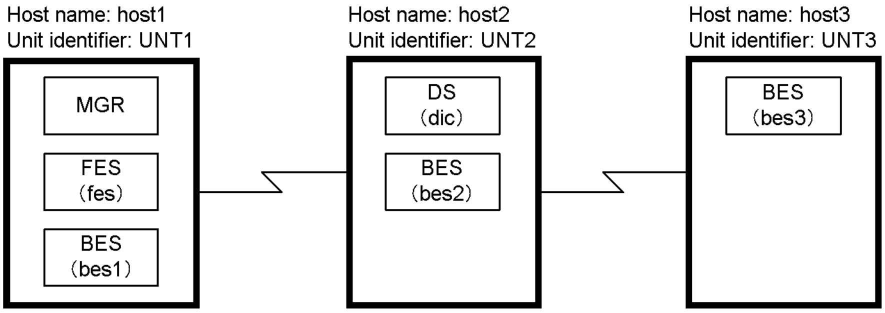
Explanation

1. Specifies the unit identifier.
2. Specifies the host name.

C.4 Example 4 (HiRDB/Parallel Server)

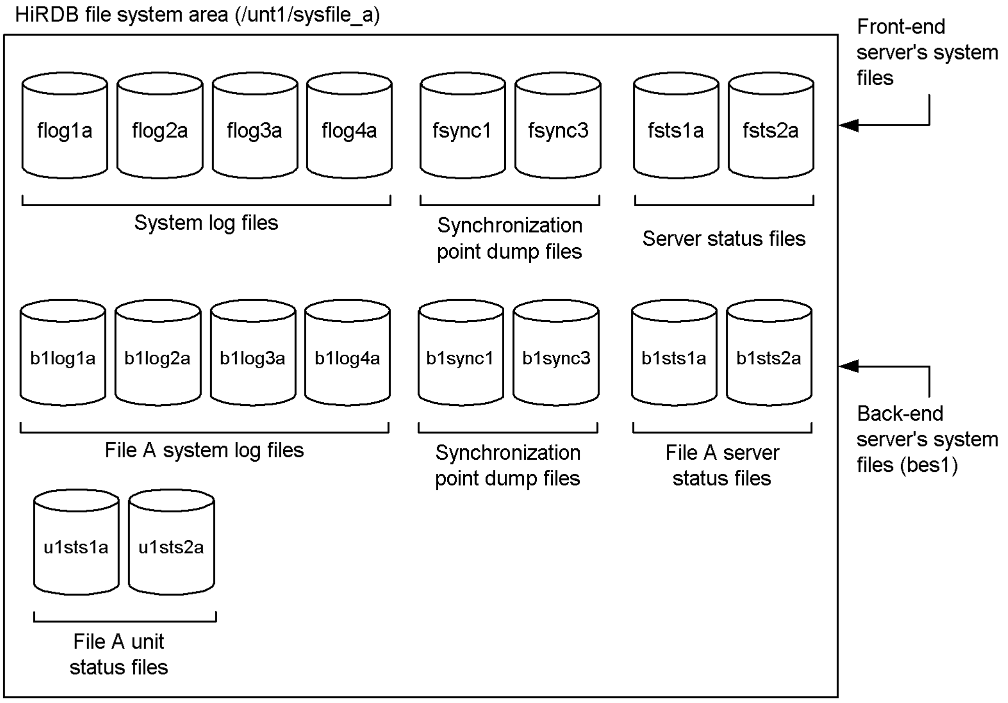
This example creates system definitions for HiRDB/Parallel Server. The system configuration is shown as follows:

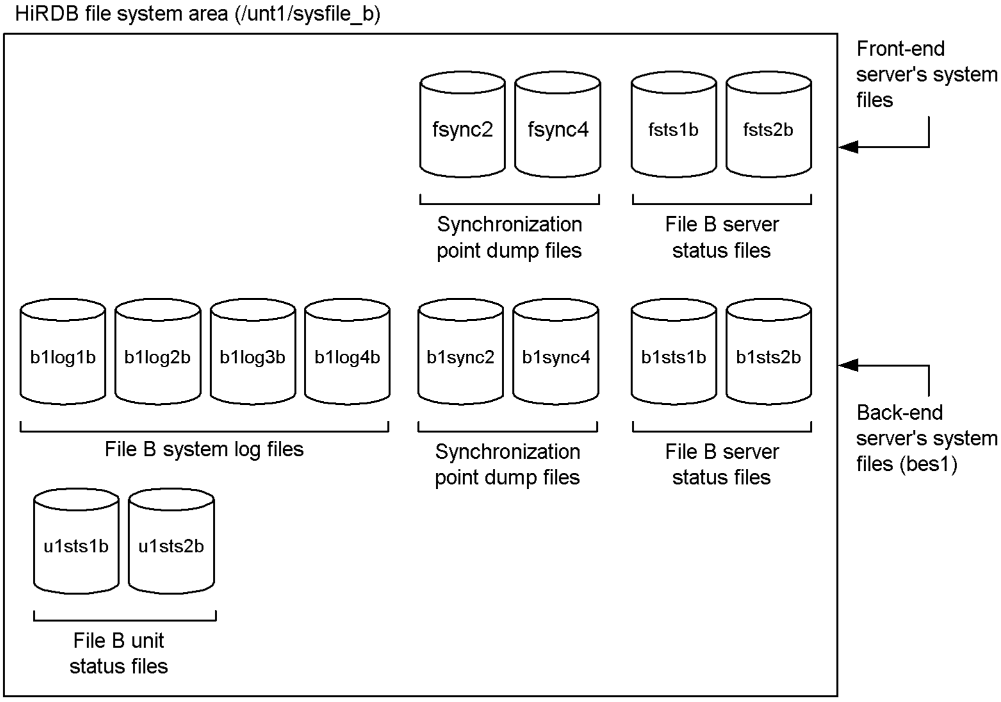
Unit configuration



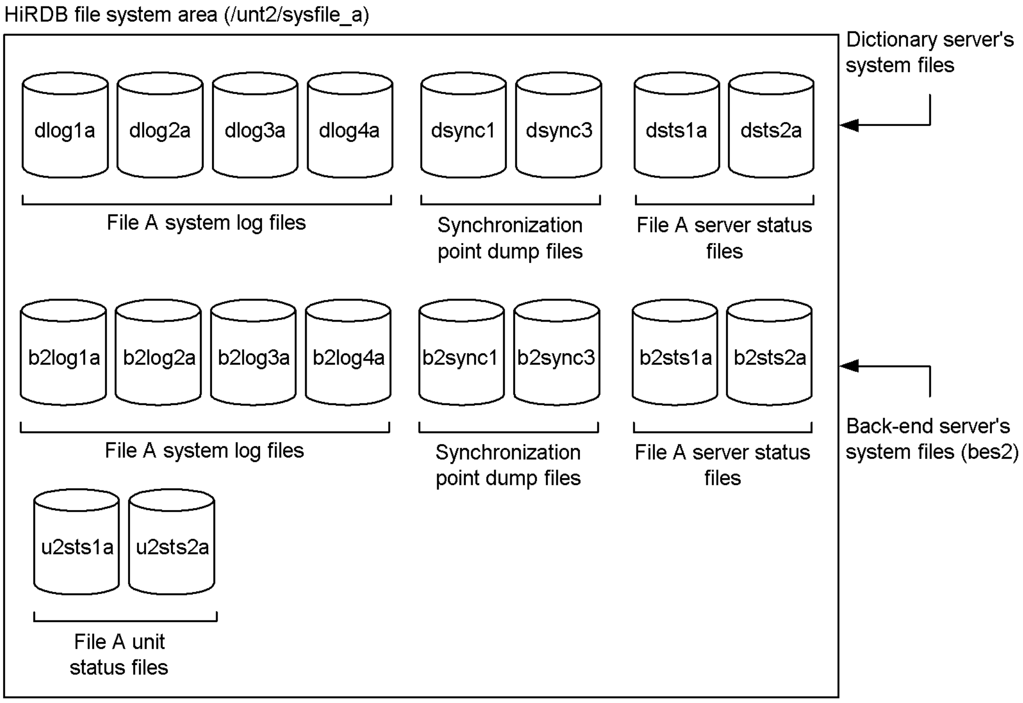
Note: Server name is shown in parentheses.

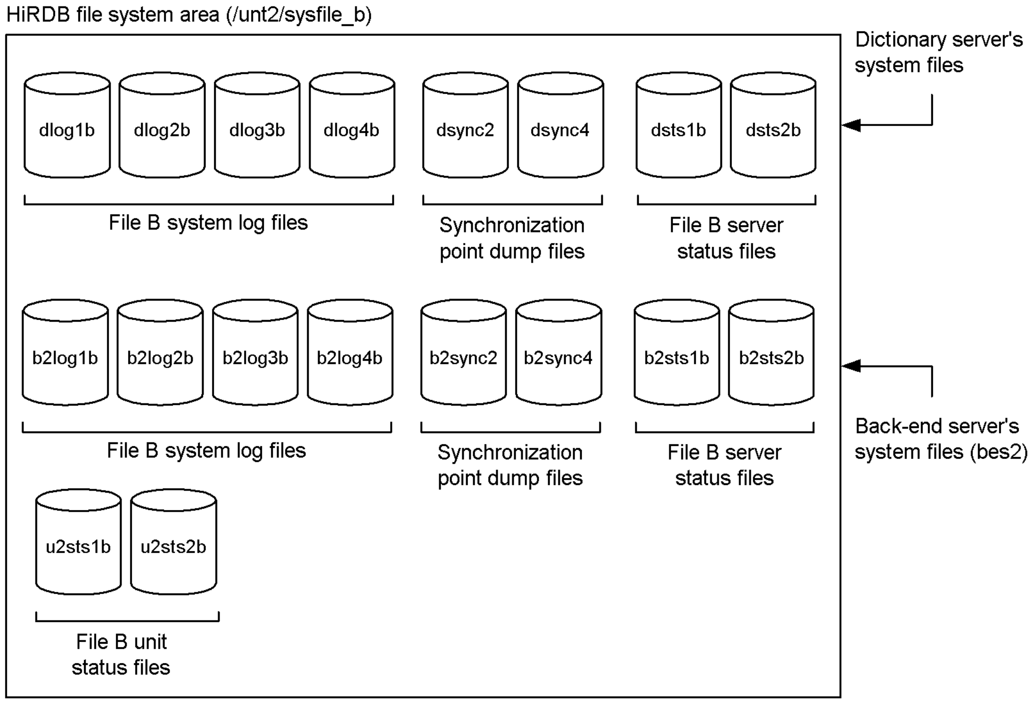
System files configuration for UNT1


HiRDB file system area (/unt1/sysfile_a)
flog1a
flog2a
flog3a
flog4a
fsync1
fsync3
fsts1a
fsts2a
System log files
Synchronization point dump files
Server status files
Front-end server's system files
b1log1a
b1log2a
b1log3a
b1log4a
b1sync1
b1sync3
b1sts1a
b1sts2a
File A system log files
Synchronization point dump files
File A server status files
Back-end server's system files (bes1)
u1sts1a
u1sts2a
File A unit status files

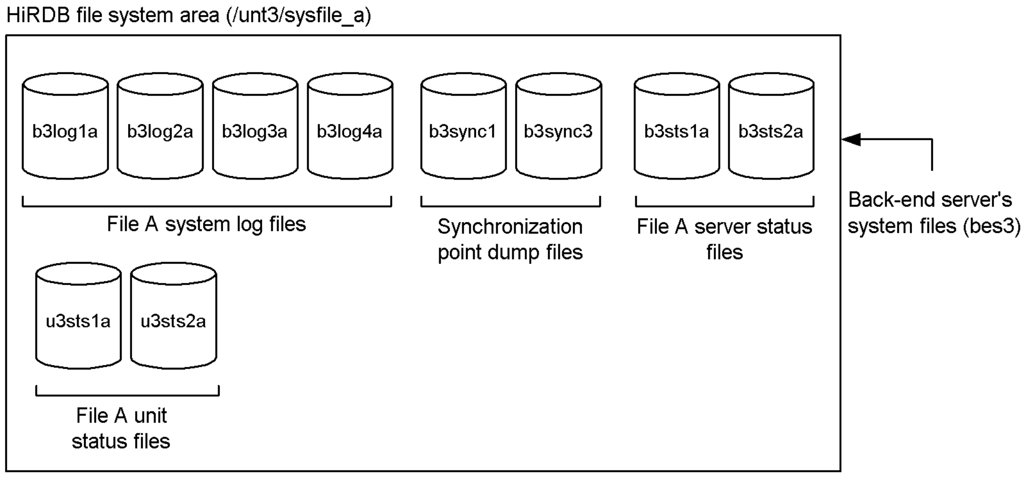

HiRDB file system area (/unt1/sysfile_b)
fsync2
fsync4
fsts1b
fsts2b
Synchronization point dump files
File B server status files
Front-end server's system files
b1log1b
b1log2b
b1log3b
b1log4b
b1sync2
b1sync4
b1sts1b
b1sts2b
File B system log files
Synchronization point dump files
File B server status files
Back-end server's system files (bes1)
u1sts1b
u1sts2b
File B unit status files

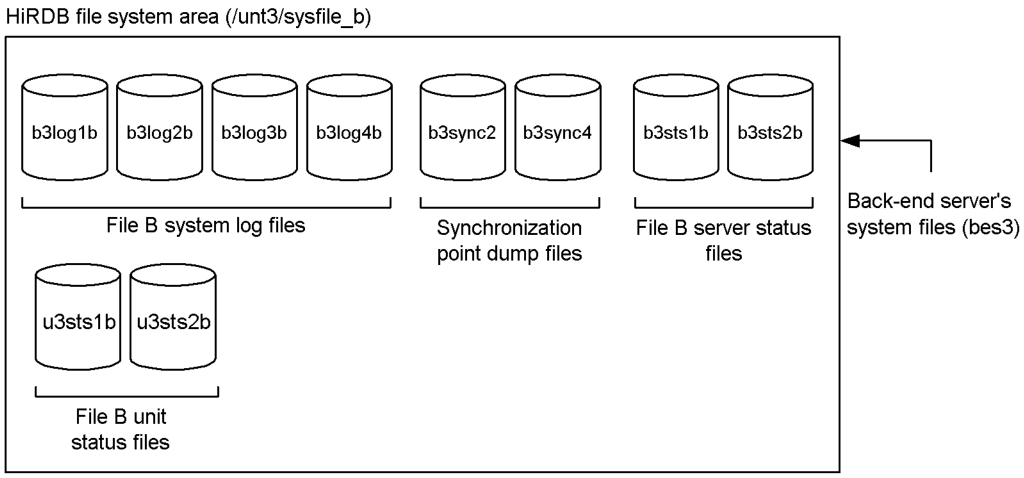
System files configuration for UNT2


HiRDB file system area (/unt2/sysfile_a)
Dictionary server's system files
dlog1a
dlog2a
dlog3a
dlog4a
dsync1
dsync3
dsts1a
dsts2a
File A system log files
Synchronization point dump files
File A server status files
b2log1a
b2log2a
b2log3a
b2log4a
b2sync1
b2sync3
b2sts1a
b2sts2a
File A system log files
Synchronization point dump files
File A server status files
Back-end server's system files (bes2)
u2sts1a
u2sts2a
File A unit status files


HiRDB file system area (/unt2/sysfile_b)
dlog1b
dlog2b
dlog3b
dlog4b
dsync2
dsync4
dsts1b
dsts2b
Dictionary server's system files
File B system log files
Synchronization point dump files
File B server status files
b2log1b
b2log2b
b2log3b
b2log4b
b2sync2
b2sync4
b2sts1b
b2sts2b
Back-end server's system files (bes2)
File B system log files
Synchronization point dump files
File B server status files
u2sts1b
u2sts2b
File B unit status files

System files configuration for UNT3


HiRDB file system area (/unt3/sysfile_a)
b3log1a
b3log2a
b3log3a
b3log4a
b3sync1
b3sync3
b3sts1a
b3sts2a
Back-end server's system files (bes3)
File A system log files
Synchronization point dump files
File A server status files
u3sts1a
u3sts2a
File A unit status files


HiRDB file system area (/unt3/sysfile_b)
b3log1b
b3log2b
b3log3b
b3log4b
b3sync2
b3sync4
b3sts1b
b3sts2b
File B system log files
Synchronization point dump files
File B server status files
u3sts1b
u3sts2b
File B unit status files
Back-end server's system files (bes3)

(1) System common definition

```
set pd_system_id = PDB1                                             1
set pd_name_port = 20001                                            2
set pd_master_file_name = "/rdarea/mast/mast01"                     3
set pd_max_users = 100                                              4
set pd_max_server_process = 520                                     5
set pd_max_access_tables = 50                                       6
set pd_sql_object_cache_size = 3000                                 7
set pd_max_rdarea_no = 200                                          8
set pd_max_file_no = 600                                            9
set pd_optimize_level = "PRIOR_NEST_JOIN","PRIOR_OR_INDEXES", \     10
    "SORT_DATA_BES","DETER_AND_INDEXES","RAPID_GROUPING", \
    "DETER_WORK_TABLE_FOR_UPDATE", \
    "APPLY_ENHANCED_KEY_COND"
set pd_additional_optimize_level = "COST_BASE_2"                    11
pdunit -x host1 -u UNT1 -d "/HiRDB"                                 12
pdunit -x host2 -u UNT2 -d "/HiRDB"
pdunit -x host3 -u UNT3 -d "/HiRDB"
pdstart -t MGR -u UNT1                                              13
pdstart -t FES -s fes -u UNT1
pdstart -t DIC -s dic -u UNT2
pdstart -t BES -s bes1 -u UNT1
pdstart -t BES -s bes2 -u UNT2
pdstart -t BES -s bes3 -u UNT3
pdbuffer -a gbuf01 -r RDMAST,RDDIC,RDDIR -n 1000                    14
pdbuffer -a gbuf02 -r RDAREA1,RDAREA2,RDAREA3 -n 1000               15
pdbuffer -a gbuf03 -r RDAREA4,RDAREA5,RDAREA6 -n 1000               16
pdbuffer -a gbuf04 -o -n 1000                                       17
putenv SHMMAX 16                                                    18
```

Explanation

1. Specifies the HiRDB identifier.
2. Specifies the HiRDB port number.
3. Specifies the first HiRDB file in the master directory RDAREA.
4. Specifies the maximum number of concurrent connections.
5. Specifies the maximum number of server processes that can be activated concurrently.
6. Specifies the maximum number of base tables that can be accessed concurrently.
7. Specifies the SQL object buffer length.
8. Specifies the maximum number of RDAREAs allowed.
9. Specifies the maximum number of HiRDB files that comprise an RDAREA.
10. Specifies SQL optimization options.
11. Specifies an SQL extension optimizing option.
12. Specifies the configurations of the units in the HiRDB/Parallel Server:

    `-x`: Host name

    `-u`: Unit identifier

    `-d`: HiRDB directory name
13. Specifies the configurations of the servers in the HiRDB/Parallel Server:

    `-t`: Server type

    `-s`: Server name

    `-u`: Unit identifier
14. Allocates the master directory RDAREA, data dictionary RDAREA, and data directory RDAREA to a global buffer.
15. Allocates user RDAREAs (RDAREA1-RDAREA3) to a global buffer.
16. Allocates user RDAREAs (RDAREA4-RDAREA6) to a global buffer.
17. Allocates other RDAREAs to a global buffer.
18. Specifies the maximum number of shared memory segments.

(2) Unit control information definition of UNT1

```
set pd_unit_id = UNT1                                                  1
set pd_syssts_file_name_1 = "u1sts1","/unt1/sysfile_a/u1sts1a",\       2
                                     "/unt1/sysfile_b/u1sts1b"
set pd_syssts_file_name_2 = "u1sts2","/unt1/sysfile_a/u1sts2a",\
                                     "/unt1/sysfile_b/u1sts2b"
```

Explanation

1. Specifies the unit identifier.
2. Specifies the configuration of the unit status files.

(3) Unit control information definition of UNT2

```
set pd_unit_id = UNT2                                                  1
set pd_syssts_file_name_1 = "u2sts1","/unt2/sysfile_a/u2sts1a",\       2
                                     "/unt2/sysfile_b/u2sts1b"
set pd_syssts_file_name_2 = "u2sts2","/unt2/sysfile_a/u2sts2a",\
                                     "/unt2/sysfile_b/u2sts2b"
```

Explanation

1. Specifies the unit identifier.
2. Specifies the configuration of the unit status files.

(4) Unit control information definition of UNT3

```
set pd_unit_id = UNT3                                                  1
set pd_syssts_file_name_1 = "u3sts1","/unt3/sysfile_a/u3sts1a",\       2
                                     "/unt3/sysfile_b/u3sts1b"
set pd_syssts_file_name_2 = "u3sts2","/unt3/sysfile_a/u3sts2a",\
                                     "/unt3/sysfile_b/u3sts2b"
```

Explanation

1. Specifies the unit identifier.
2. Specifies the configuration of the unit status files.

(5) Front-end server definition

```
set pd_log_dual = N                                                       1
set pd_sts_file_name_1 = "fsts1","/unt1/sysfile_a/fsts1a",\               2
                                 "/unt1/sysfile_b/fsts1b"
set pd_sts_file_name_2 = "fsts2","/unt1/sysfile_a/fsts2a",\
                                 "/unt1/sysfile_b/fsts2b"
pdlogadfg -d sys -g flog1 ONL                                             3
pdlogadfg -d sys -g flog2 ONL
pdlogadfg -d sys -g flog3 ONL
pdlogadfg -d sys -g flog4 ONL
pdlogadpf -d sys -g flog1 -a "/unt1/sysfile_a/flog1a"
pdlogadpf -d sys -g flog2 -a "/unt1/sysfile_a/flog2a"
pdlogadpf -d sys -g flog3 -a "/unt1/sysfile_a/flog3a"
pdlogadpf -d sys -g flog4 -a "/unt1/sysfile_a/flog4a"
pdlogadfg -d spd -g fsync1 ONL                                            4
pdlogadfg -d spd -g fsync2 ONL
pdlogadfg -d spd -g fsync3 ONL
pdlogadfg -d spd -g fsync4 ONL
pdlogadpf -d spd -g fsync1 -a "/unt1/sysfile_a/fsync1"
pdlogadpf -d spd -g fsync2 -a "/unt1/sysfile_b/fsync2"
pdlogadpf -d spd -g fsync3 -a "/unt1/sysfile_a/fsync3"
pdlogadpf -d spd -g fsync4 -a "/unt1/sysfile_b/fsync4"
```

Explanation

1. Specifies that dual system log files will not be used.
2. Specifies the configuration of the server status files.
3. Specifies the configuration of the system log files.
4. Specifies the configuration of the synchronization point dump files.

(6) Dictionary server definition

```
set pd_dic_shmpool_size = 5000                                            1
set pd_log_dual = Y                                                       2
set pd_sts_file_name_1 = "dsts1","/unt2/sysfile_a/dsts1a",\               3
                                 "/unt2/sysfile_b/dsts1b"
set pd_sts_file_name_2 = "dsts2","/unt2/sysfile_a/dsts2a",\
                                 "/unt2/sysfile_b/dsts2b"
pdwork -v "/unt2/work01","/unt2/work02"                                   4
pdlogadfg -d sys -g dlog1 ONL                                             5
pdlogadfg -d sys -g dlog2 ONL
pdlogadfg -d sys -g dlog3 ONL
pdlogadfg -d sys -g dlog4 ONL
pdlogadpf -d sys -g dlog1 -a "/unt2/sysfile_a/dlog1a"\
                          -b "/unt2/sysfile_b/dlog1b"
pdlogadpf -d sys -g dlog2 -a "/unt2/sysfile_a/dlog2a"\
                          -b "/unt2/sysfile_b/dlog2b"
pdlogadpf -d sys -g dlog3 -a "/unt2/sysfile_a/dlog3a"\
                          -b "/unt2/sysfile_b/dlog3b"
pdlogadpf -d sys -g dlog4 -a "/unt2/sysfile_a/dlog4a"\
                          -b "/ut2/sysfile_b/dlog4b"
pdlogadfg -d spd -g dsync1 ONL                                            6
pdlogadfg -d spd -g dsync2 ONL
pdlogadfg -d spd -g dsync3 ONL
pdlogadfg -d spd -g dsync4 ONL
pdlogadpf -d spd -g dsync1 -a "/unt2/sysfile_a/dsync1"
pdlogadpf -d spd -g dsync2 -a "/unt2/sysfile_b/dsync2"
pdlogadpf -d spd -g dsync3 -a "/unt2/sysfile_a/dsync3"
pdlogadpf -d spd -g dsync4 -a "/unt2/sysfile_b/dsync4"
```

Explanation

1. Specifies the size of the shared memory to be used by the dictionary server.
2. Specifies that dual system log files will be used.
3. Specifies the configuration of the server status files.
4. Specifies HiRDB file system areas for work table files.
5. Specifies the configuration of the system log files.
6. Specifies the configuration of the synchronization point dump files.

(7) Back-end server definition of bes1

```
set pd_bes_shmpool_size = 5000                                              1
set pd_log_dual = Y                                                         2
set pd_sts_file_name_1 = "b1sts1","/unt1/sysfile_a/b1sts1a",\               3
                                  "/unt1/sysfile_b/b1sts1b"
set pd_sts_file_name_2 = "b1sts2","/unt1/sysfile_a/b1sts2a",\
                                  "/unt1/sysfile_b/b1sts2b"
pdwork -v "/unt1/work01","/unt1/work02"                                     4
pdlogadfg -d sys -g b1log1 ONL                                              5
pdlogadfg -d sys -g b1log2 ONL
pdlogadfg -d sys -g b1log3 ONL
pdlogadfg -d sys -g b1log4 ONL
pdlogadpf -d sys -g b1log1 -a "/unt1/sysfile_a/b1log1a"\
                           -b "/unt1/sysfile_b/b1log1b"
pdlogadpf -d sys -g b1log2 -a "/unt1/sysfile_a/b1log2a"\
                           -b "/unt1/sysfile_b/b1log2b"
pdlogadpf -d sys -g b1log3 -a "/unt1/sysfile_a/b1log3a"\
                           -b "/unt1/sysfile_b/b1log3b"
pdlogadpf -d sys -g b1log4 -a "/unt1/sysfile_a/b1log4a"\
                           -b "/unt1/sysfile_b/b1log4b"
pdlogadfg -d spd -g b1sync1 ONL                                             6
pdlogadfg -d spd -g b1sync2 ONL
pdlogadfg -d spd -g b1sync3 ONL
pdlogadfg -d spd -g b1sync4 ONL
pdlogadpf -d spd -g b1sync1 -a "/unt1/sysfile_a/b1sync1"
pdlogadpf -d spd -g b1sync2 -a "/unt1/sysfile_b/b1sync2"
pdlogadpf -d spd -g b1sync3 -a "/unt1/sysfile_a/b1sync3"
pdlogadpf -d spd -g b1sync4 -a "/unt1/sysfile_b/b1sync4"
```

Explanation

1. Specifies the size of the shared memory to be used by the back-end server (bes1).
2. Specifies that dual system log files will be used.
3. Specifies the configuration of the server status files.
4. Specifies HiRDB file system areas for work table files.
5. Specifies the configuration of the system log files.
6. Specifies the configuration of the synchronization point dump files.

(8) Back-end server definition of bes2

```
set pd_bes_shmpool_size = 5000                                                1
set pd_log_dual = Y                                                           2
set pd_sts_file_name_1 = "b2sts1","/unt2/sysfile_a/b2sts1a",\                 3
                                   "/unt2/sysfile_b/b2sts1b"
set pd_sts_file_name_2 = "b2sts2","/unt2/sysfile_a/b2sts2a",\
                                   "/unt2/sysfile_b/b2sts2b"
pdwork -v "/unt2/work03","/unt2/work04"                                       4
pdlogadfg -d sys -g b2log1 ONL                                                5
pdlogadfg -d sys -g b2log2 ONL
pdlogadfg -d sys -g b2log3 ONL
pdlogadfg -d sys -g b2log4 ONL
pdlogadpf -d sys -g b2log1 -a "/unt2/sysfile_a/b2log1a"\
                           -b "/unt2/sysfile_b/b2log1b"
pdlogadpf -d sys -g b2log2 -a "/unt2/sysfile_a/b2log2a"\
                           -b "/unt2/sysfile_b/b2log2b"
pdlogadpf -d sys -g b2log3 -a "/unt2/sysfile_a/b2log3a"\
                           -b "/unt2/sysfile_b/b2log3b"
pdlogadpf -d sys -g b2log4 -a "/unt2/sysfile_a/b2log4a"\
                           -b "/unt2/sysfile_b/b2log4b"
pdlogadfg -d spd -g b2sync1 ONL                                               6
pdlogadfg -d spd -g b2sync2 ONL
pdlogadfg -d spd -g b2sync3 ONL
pdlogadfg -d spd -g b2sync4 ONL
pdlogadpf -d spd -g b2sync1 -a "/unt2/sysfile_a/b2sync1"
pdlogadpf -d spd -g b2sync2 -a "/unt2/sysfile_b/b2sync2"
pdlogadpf -d spd -g b2sync3 -a "/unt2/sysfile_a/b2sync3"
pdlogadpf -d spd -g b2sync4 -a "/unt2/sysfile_b/b2sync4"
```

Explanation

1. Specifies the size of the shared memory to be used by the back-end server (bes2).
2. Specifies that dual system log files will be used.
3. Specifies the configuration of the server status files
4. Specifies HiRDB file system areas for work table files.
5. Specifies the configuration of the system log files.
6. Specifies the configuration of the synchronization point dump files.

(9) Back-end server definition of bes3

```
set pd_bes_shmpool_size = 5000                                              1
set pd_log_dual = Y                                                         2
set pd_sts_file_name_1 = "b3sts1","/unt3/sysfile_a/b3sts1a",\               3
                                  "/unt3/sysfile_b/b3sts1b"
set pd_sts_file_name_2 = "b3sts2","/unt3/sysfile_a/b3sts2a",\
                                  "/unt3/sysfile_b/b3sts2b"
pdwork -v "/unt3/work01","/unt3/work02"                                     4
pdlogadfg -d sys -g b3log1 ONL                                              5
pdlogadfg -d sys -g b3log2 ONL
pdlogadfg -d sys -g b3log3 ONL
pdlogadfg -d sys -g b3log4 ONL
pdlogadpf -d sys -g b3log1 -a "/unt3/sysfile_a/b3log1a"\
                           -b "/unt3/sysfile_b/b3log1b"
pdlogadpf -d sys -g b3log2 -a "/unt3/sysfile_a/b3log2a"\
                           -b "/unt3/sysfile_b/b3log2b"
pdlogadpf -d sys -g b3log3 -a "/unt3/sysfile_a/b3log3a"\
                           -b "/unt3/sysfile_b/b3log3b"
pdlogadpf -d sys -g b3log4 -a "/unt3/sysfile_a/b3log4a"\
                           -b "/unt3/sysfile_b/b3log4b"
pdlogadfg -d spd -g b3sync1 ONL                                             6
pdlogadfg -d spd -g b3sync2 ONL
pdlogadfg -d spd -g b3sync3 ONL
pdlogadfg -d spd -g b3sync4 ONL
pdlogadpf -d spd -g b3sync1 -a "/unt3/sysfile_a/b3sync1"
pdlogadpf -d spd -g b3sync2 -a "/unt3/sysfile_b/b3sync2"
pdlogadpf -d spd -g b3sync3 -a "/unt3/sysfile_a/b3sync3"
pdlogadpf -d spd -g b3sync4 -a "/unt3/sysfile_b/b3sync4"
```

Explanation

1. Specifies the size of the shared memory to be used by the back-end server (bes3).
2. Specifies that dual system log files will be used.
3. Specifies the configuration of the server status files.
4. Specifies HiRDB file system areas for work table files.
5. Specifies the configuration of the system log files.
6. Specifies the configuration of the synchronization point dump files.

(10) UAP environment definition

```
set pd_uap_wait = Y                                         1
pdlbuffer -a localbuf1 -r RDAREA10 -n 1000 -p 16            2
pdlbuffer -a localbuf2 -r RDAREA11,RDAREA12 -n 1000         3
pdlbuffer -a localbuf3 -i USER01.INDX01 -n 1000             4
```

Explanation

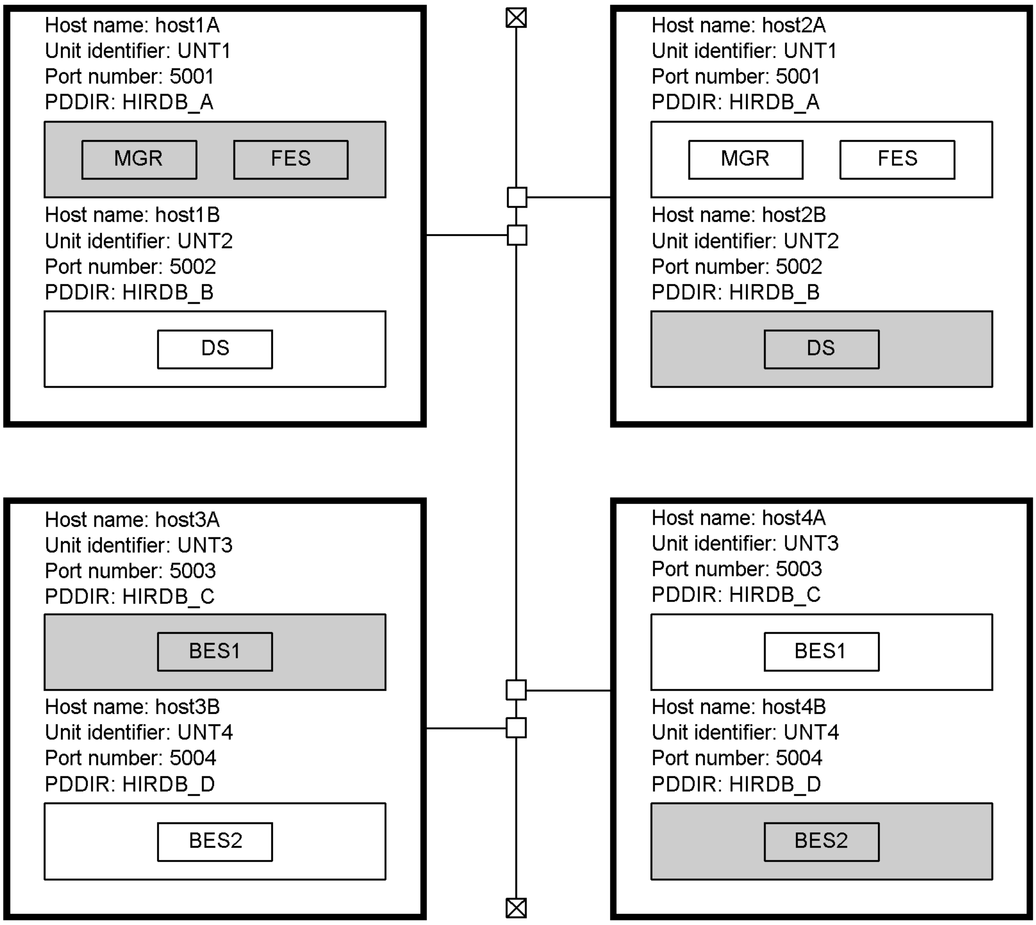
1. Specifies the action to be taken by the UAP when the RDAREA or index to be accessed using a local buffer is being used by another user.
2. Allocates a local buffer to a user RDAREA (`RDAREA10`).
3. Allocates a local buffer to user RDAREAs (`RDAREA11` and `RDAREA12`).
4. Allocates a local buffer to an index (`INDX01`).

C.5 Example 5 (HiRDB/Parallel Server: when the standby system switchover facility is used)

This example creates system definitions for a HiRDB/Parallel Server. The system configuration is shown as follows.

- Mutual system switchover is used.
- IP addresses are not inherited.
- The rapid system switchover facility is used for all units.

Note that in this definition example, only those operands that are related to the system common definition and unit control information definition are explained.



Note: Shading indicates the primary HiRDB systems.

Key points

- When IP addresses are not inherited, specify a host name using an alias IP address for the `-x` and `-c` options of the `pdunit` operand, and not the standard host name, to maintain independence from the server machine.
- When mutual system switchover without IP address inheriting is used, the host name to be specified for the `-x` and `-c` options of the `pdunit` operand must be unique. See the `pdunit` operand specification example in (*1*).

(1) System common definition

```
         :
set pd_name_port = 5001                                          1
         :
set pd_ha = use                                                  2
set pd_ha_ipaddr_inherit = N
set pd_ha_switch_timeout = Y
set pd_ha_transaction = queuing
set pd_ha_trn_queuing_wait_time = 240
set pd_ha_trn_restart_retry_time = 90
         :
pdunit -x host1A -u UNT1 -d "/HiRDB_A" -c host2A -p 5001         3
pdunit -x host2B -u UNT2 -d "/HiRDB_B" -c host1B -p 5002
pdunit -x host3A -u UNT3 -d "/HiRDB_C" -c host4A -p 5003
pdunit -x host4B -u UNT4 -d "/HiRDB_D" -c host3B -p 5004
pdstart -t MGR -u UNT1                                           4
pdstart -t FES -s fes1 -u UNT1
pdstart -t DIC -s dic  -u UNT2
pdstart -t BES -s bes1 -u UNT3
pdstart -t BES -s bes2 -u UNT4
         :
```

Explanation

1. Specifies the HiRDB port number.
2. Specifies the operands related to the system switchover facility.
3. Specifies the unit configuration of HiRDB/Parallel Server:

   `-x`: Host name

   `-u`: Unit identifier

   `-d`: HiRDB directory name

   `-c`: Host name of the secondary system

   `-p`: Port number of the unit
4. Specifies the server configuration of the HiRDB/Parallel Server:

   `-t`: Server type

   `-s`: Server name

   `-u`: Unit identifier

(2) Unit control information definitions for UNT1-UNT4

```
         :
set pd_ha_acttype = server                        1
set pd_ha_agent = standbyunit                     2
         :
```

Explanation

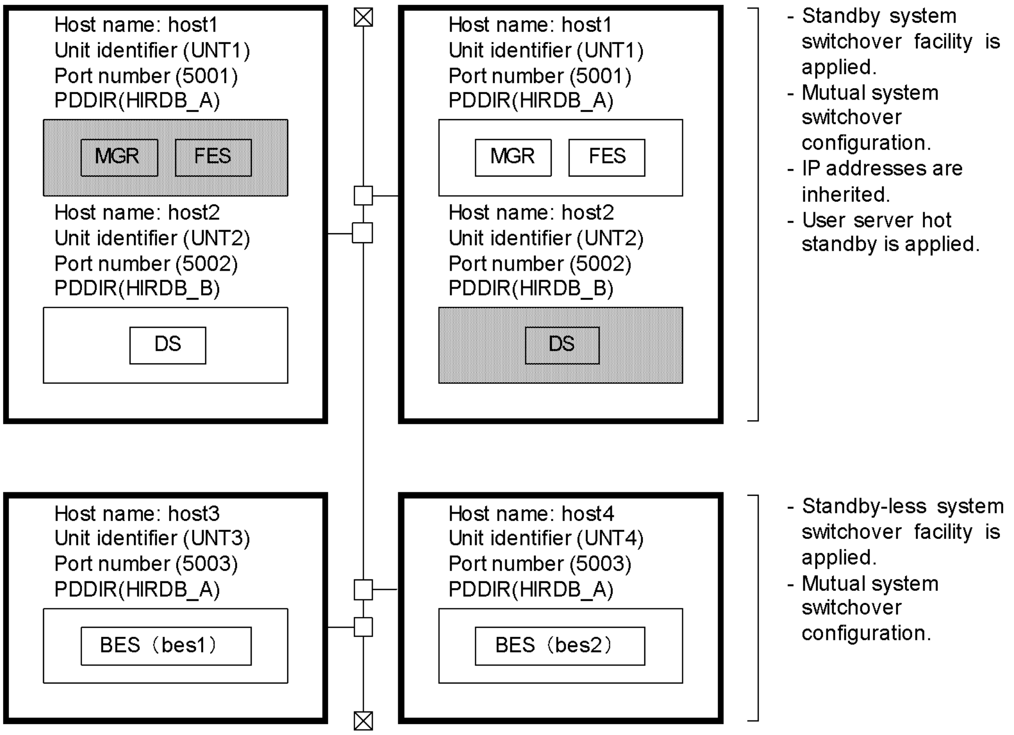
1. Specifies that the system switchover facility be used in the server mode.
2. Specifies that the rapid system switchover facility be used.

C.6 Example 6 (HiRDB/Parallel Server: when the standby-less system switchover (1:1) facility is used)

This example creates system definitions for a HiRDB/Parallel Server. The system configuration is shown as follows.

- Both standby and standby-less system switchover facilities are used.
- The units to which the standby system switchover facility is applied are based on a mutual system switchover configuration and inherit IP addresses. User server hot standby is also used.
- The units to which the standby-less system switchover (1:1) facility is applied are based on a mutual alternating configuration.

Note that in this definition example, only those operands that are related to the system common definition and unit control information definition are explained.



Note

- Shading indicates the primary units.
- Server name is shown in parentheses.
- Back-end server (bes1) is a normal BES and is also an alternate BES for back-end server (bes2).
- Back-end server (bes2) is a normal BES and is also an alternate BES for back-end server (bes1).

(1) System common definition

```
         :
set pd_name_port = 5001                                          1
         :
set pd_ha = use                                                  2
set pd_ha_ipaddr_inherit = Y
set pd_ha_switch_timeout = Y
set pd_ha_transaction = queuing
set pd_ha_trn_queuing_wait_time = 240
set pd_ha_trn_restart_retry_time = 90
         :
pdunit -x host1 -u UNT1 -d "/HiRDB_A" -p 5001                    3
pdunit -x host2 -u UNT2 -d "/HiRDB_B" -p 5002
pdunit -x host3 -u UNT3 -d "/HiRDB_A" -p 5003
pdunit -x host4 -u UNT4 -d "/HiRDB_A" -p 5003
pdstart -t MGR -u UNT1                                           4
pdstart -t FES -s fes1 -u UNT1
pdstart -t DIC -s dic  -u UNT2
pdstart -t BES -s bes1 -u UNT3 -c bes2                           5
pdstart -t BES -s bes2 -u UNT4 -c bes1
         :
```

Explanation

1. Specifies the HiRDB port number.
2. Specifies the operands related to the system switchover facility.
3. Specifies the unit configuration of HiRDB/Parallel Server:

   -x: Host name

   -u: Unit identifier

   -d: HiRDB directory name

   -p: Port number of the unit
4. Specifies the server configuration of the HiRDB/Parallel Server:

   -t: Server type

   -s: Server name

   -t: Server type

5. Specifies the server configuration of the HiRDB/Parallel Server:

   `-t`: Server type

   `-s`: Specifies the name of the normal BES.

   `-t`: Server type

   `-c`: Specifies the name of the alternate BES.

(2) Unit control information definitions for UNT1 and UNT2

```
         :
set pd_ha_acttype = server                                        1
set pd_ha_server_process_standby = Y                              2
         :
```

Explanation

1. Specifies that the system switchover facility be used in the server mode.
2. Specifies that user server hot standby be used.

(3) Unit control information definitions for UNT3 and UNT4

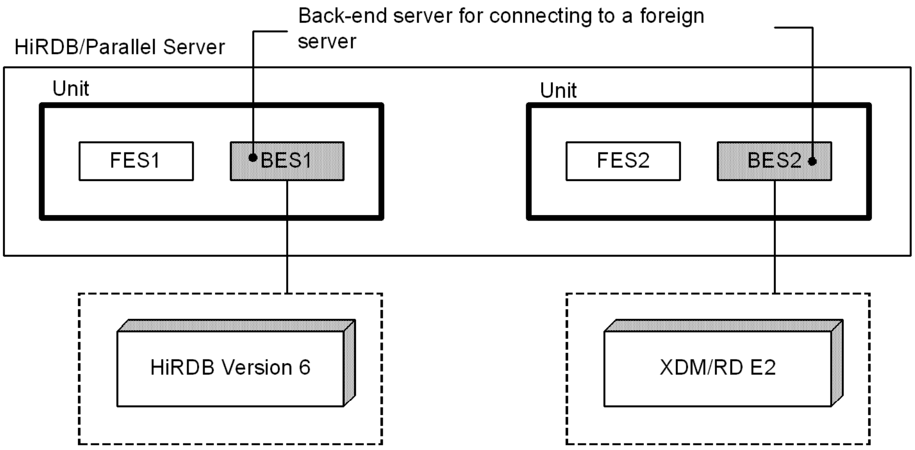
```
         :
set pd_ha_acttype = server                                        1
set pd_ha_agent = server                                          2
         :
```

Explanation

1. Specifies that the system switchover facility be used in the server mode.
2. Specifies that the standby-less system switchover (1:1) facility be used.

C.7 Example 7 (when the HiRDB External Data Access facility is used)

This example creates system definitions for a HiRDB/Parallel Server when the HiRDB External Data Access is being used. Note that in this definition example, only those operands that are related to the HiRDB External Data Access facility are explained.



(1) System common definition

```
        :
set pd_max_foreign_server = 2
```

Explanation

Specifies the (maximum) number of foreign servers.

(2) Front-end server definition (FES1)

```
        :
pdhubopt -s hirdb_v6_01 -f hubopt1
pdhubopt -s xdmrd_01 -f hubopt2
```

Explanation

Defines that HiRDB Version 6 (server name: `hirdb_v6_01`) and XDM/RD E2 (server name: `xdmrd_01`) are used as foreign servers. A Hub optimization information definition file name is specified for the `-f` option.

(3) Front-end server definition (FES2)

```
        :
pdhubopt -s hirdb_v6_01 -f hubopt1
pdhubopt -s xdmrd_01 -f hubopt2
```

Explanation

Defines that HiRDB Version 6 (server name: `hirdb_v6_01`) and XDM/RD E2 (server name: `xdmrd_01`) are used as foreign servers. A Hub optimization information definition file name is specified for the `-f` option.

(4) Back-end server definition (BES1)

```
         :
set pd_foreign_server_libpath = "/HiRDB/client/lib"
```

Explanation

Specifies the absolute path name of the client library of the foreign server (HiRDB Version 6).

(5) Back-end server definition (BES2)

```
         :
set pd_foreign_server_libpath = "/HiRDB/client/lib"
```

Explanation

Specifies the absolute path name of the client library of the foreign server (XDM/RD E2).

(6) Foreign server information definition (for HiRDB Version 6)

```
putenv PDHOST host01                                            1
putenv PDNAMEPORT 20000                                         2
```

Explanation

1. Specifies the host name of the foreign server (HiRDB Version 6).
2. Specifies the port number of the foreign server (HiRDB Version 6).

(7) Foreign server information definition (for XDM/RD E2)

```
putenv PDHOST host02                                            1
putenv PDNAMEPORT 20100                                         2
putenv PDCLTRDNODE rd_node1                                     3
```

Explanation

1. Specifies the host name of the DB connection server.
2. Specifies the port number of the DB connection server.
3. Specifies the RD node name of the foreign server (XDM/RD E2).

(8) Hub optimization information definition (for HiRDB Version 6)

Hub optimization information definition file name: `hubopt1`

```
set pd_hub_opt_on_cnd =1
set pd_hub_opt_joined_table =1
set pd_hub_opt_set_func = 1
set pd_hub_opt_case = 1
set pd_hub_opt_like = 1
set pd_hub_opt_grouping = 1
set pd_hub_opt_data_len = 255
set pd_hub_opt_abs = 1
set pd_hub_opt_date = 1
set pd_hub_opt_time = 1
set pd_hub_opt_digits = 1
set pd_hub_opt_length = 1
set pd_hub_opt_lower_upper_type = 1
set pd_hub_opt_mod_div_type = 1
set pd_hub_opt_substr = 1
set pd_hub_opt_num = 1
set pd_hub_opt_datetime = 1
set pd_hub_opt_datetime_op = 1
set pd_hub_opt_trailing_spc = 1
set pd_hub_opt_nullable = 1
set pd_hub_opt_use_zero_string = 1
set pd_hub_opt_nchar = 1
set pd_hub_opt_nest_scalar = 0
set pd_hub_opt_float = 2
set pd_hub_opt_table_num = 64
set pd_hub_opt_time_24hour = 1
```

(9) Hub optimization information definition (for XDM/RD E2)

Hub optimization information definition file name: `hubopt2`

```
set pd_hub_opt_on_cnd =1
set pd_hub_opt_joined_table =1
set pd_hub_opt_set_func = 3
set pd_hub_opt_case = 1
set pd_hub_opt_like = 4
set pd_hub_opt_grouping = 4
set pd_hub_opt_col_len = 255
set pd_hub_opt_abs = 1
set pd_hub_opt_date = 1
set pd_hub_opt_time = 1
set pd_hub_opt_digits = 1
set pd_hub_opt_length = 3
set pd_hub_opt_lower_upper_type = 1
set pd_hub_opt_mod_div_type = 2
set pd_hub_opt_substr = 3
set pd_hub_opt_num = 1
set pd_hub_opt_datetime = 1
set pd_hub_opt_datetime_op = 1
set pd_hub_opt_trailing_spc = 1
set pd_hub_opt_nullable = 1
set pd_hub_opt_use_zero_string = 1
set pd_hub_opt_nchar = 2
set pd_hub_opt_nest_scalar = 0
set pd_hub_opt_float = 2
set pd_hub_opt_table_num = 64
set pd_hub_opt_time_24hour = 1
```

D. Formulas for Determining Operand Specification Values

This appendix provides and explains formulas for determining specification values for various system definition operands. This appendix contains the following sections:

1. Formulas for determining size of statistics log file (`pd_stj_file_size`)
2. Formulas for determining size of SQL object buffer (`pd_sql_object_cache_size`)
3. Formulas for determining size of table definition information buffer (`pd_table_def_cache_size`)
4. Formulas for determining total number of tables and RDAREAs per server locked with `UNTIL DISCONNECT` specified (`pd_lck_until_disconnect_cnt`)
5. Formulas for determining size of routine definition information buffer (`pd_routine_def_cache_size`)

D.1 Formulas for determining size of statistics log file (pd_stj_file_size)

The size of the statistics log that is output differs depending on the statistical information. The size of the statistics log for each type of statistical information is described starting in (1) as follows. For the `pd_stj_file_size` operand, specify a value that is equal to or greater than the combined total of the statistics log sizes determined in (1) and beyond.

Note

Each of these formulas produces a size in bytes; each result in bytes should be converted subsequently to kilobytes.

(1) Statistical information related to system operation (sys)

This will depend on the options specified in the `pdstbegin` operand or `pdstbegin` command.

When neither -a nor -s option is specified

Statistics log size = 2412 × ($\downarrow a \div b \downarrow$ + *number-times-pdstjsync-command-is-executed-during-statistical-information-output-time*) (Bytes)

When -a option is specified

Statistics log size = (1836 × c + 2412) × ($\downarrow a \div b \downarrow$ + *number-times-pdstjsync-command-is-executed-during-statistical-information-output-time*) (Bytes)

When -s option is specified

Statistics log size = $(1836 \times d + 2412) \times (\downarrow a \div b \downarrow +$ *number-times-pdstjsync-command-is-executed-during-statistical-information-output-time*) (Bytes)

a:

Statistical information output period (minutes)

b:

Statistical information output interval (minutes)

Interval specified in the `pdstbegin` operand or the `-m` option of the `pdstbegin` command

c:

Number of servers in unit

System manager is not included.

d:

Number of servers specified in the `pdstbegin` operand or the `-s` option of the `pdstbegin` command

(2) Statistical information related to UAPs (uap)

Formula

Statistics log size = 1040 × *number-of-UAPS-to-be-executed-during-statistical-information-collection* (Bytes)

If the total number of UAPs that will actually be executed is known, use the formula above. If the number of UAPs that will be executed during statistical information collection is not known, that number can be determined with the following formula:

- *Number of UAPs* = *number-of-UAPS-to-be-executed-per-unit-time* × *statistical-information-collection-period* ÷ *unit-time*

For example, if 10 UAPs are executed in 30 minutes and statistical information is collected for a period of 60 minutes, the UAP count will be $10 \times 60 \div 30 = 20$.

(3) Statistical information related to SQL (sql)

Formula

Statistics log size = 728 × *number-of-SQLs-to-be-executed-during-statistical-information-collection** (Bytes)

If the total number of SQLs that will actually be executed is known, use the formula above. If the number of SQLs that will be executed during statistical information collection is not known, that number can be determined with the following formula:

Number of SQLs[*] = *average-number-of-SQL-statements-executed by-UAP* × (*number-of-UAPs-to-be-executed-per-unit-time* × *statistical-information-collection-period* ÷ *unit-time*)

[*] If a stored procedure or stored function will be executed, the SQL statements in the procedure or function must also be counted.

(4) Statistical information related to global buffers (buf)

Formula

Statistics log size = 380 × *a* × *b* (Bytes)

a:

Number specified by the `pdbuffer` operand

b:

Number of times synchronization point dump is collected during statistical information collection

The number of times a synchronization point dump is collected during the statistical information collection period can be determined with the following formula:

↑
size-of-system-log-output-during-statistical-information-collection-period

÷

(*total-of-pd_log_max_data_size-operand-values-in-individual-server-definitions* ÷ 3)

×

total-of-pd_log_sdinterval-operand-specification-values-in-individual-server-definitions ↑

For details about the system log size, see the *HiRDB Version 8 Installation and Design Guide*.

(5) HiRDB file statistical information related to database manipulation (fil)

Formula

Statistics log size = 428 × *a* × *b* (Bytes)

a:

Number of HiRDB files specified in the control statement of the database initialization utility

b:

Number of times synchronization point dump is collected during statistical information collection

The number of times a synchronization point dump is collected during the statistical information collection period can be determined with the following formula:

↑
size-of-system-log-size-output-during-statistical-information-collection-period

÷
(*total-of-pd_log_max_data_size-operand-specification-values-in-individual-server-definitions* ÷ 3)

×
total-of-pd_log_sdinterval-operand-specification-values-in-individual-server-definitions ↑

For details about the system log size, see the *HiRDB Version 8 Installation and Design Guide*.

(6) Statistical information related to deferred write processing (dfw)

Formula

Statistics log size = $384 \times a \times 2 + (384 \times b \times c) \div (d \times e \times f)$ (Bytes)

a:

Number of times synchronization point dump is collected during statistical information collection

The number of times a synchronization point dump is collected during the statistical information collection period can be determined with the following formula:

↑
size-of-system-log-size-output-during-statistical-information-collection-period

÷
(*total-of-pd_log_max_data_size-operand-specification-values-in-individual-server-definitions* ÷ 3)

×
total-of-pd_log_sdinterval-operand-specification-values-in-individual-serv

er-definitions $\uparrow$

For details about the system log size, see the *HiRDB Version 8 Installation and Design Guide*.

b:

Average number of pages updated per transaction

c:

Number of transactions to be executed within statistical information collection period

d:

Total buffer sectors count specified by the `-n` option of the `pdbuffer` operand

e:

Database-updating transactions as a percentage of all transactions

f:

Updated pages output ratio during deferred write, as specified by the `pdbuffer` operand

(7) Statistical information related to indexes (idx)

Formula

Statistics log size = $3768 \times \uparrow a \div 128 \uparrow \times b$ (Bytes)

a:

Total number of partitioned indexes

The total number of partitioned indexes is the sum of the numbers of RDAREA names specified by IN RDAREA of individual `CREATE INDEX` statements of definition SQLs.

b:

Number of times synchronization point dump is collected during statistical information collection

The number of times a synchronization point dump is collected during the statistical information collection period can be determined with the following formula:

$\uparrow$
size-of-system-log-output-during-statistical-information-collection-period

$\div$

(*total-of-pd_log_max_data_size-operand-specification-values-in-individual-server-definitions* ÷ 3)

×

total-of-pd_log_sdinterval-operand-specification-values-in-individual-server-definitions ↑

For details about the system log size, see the *HiRDB Version 8 Installation and Design Guide*.

(8) Statistical information related to SQL static optimization (sop)

Formula

Statistics log size = 92 × *a* (Bytes)

a:

Number of SQL objects to be created in the SQL statements to be executed during statistical information collection. This applies to the following SQL statements: `PREPARE` statement, `EXECUTE IMMEDIATE` statement, and static SQL statements.

If it is possible to determine the actual number of SQL objects to be created, this formula can be used to make that determination. If the number of SQL objects to be created during statistical information collection is not known, the number of SQL objects to be created can be determined using the following formula:

Number of SQL objects to be created = average-number-of-SQL-statements-to-be-executed-in-UAP × *SQL-object-cache-miss-rate* × *(number-of-UAP-executions-per-unit-of-time* × *statistical-information-collection-time* ÷ *unit-of-time)*

The SQL object cache miss rate can be determined from the statistical information related to the system.

(9) Statistical information related to SQL dynamic optimization (dop)

Formula

Statistics log size =

$$\sum_{i=1}^{a} \left\{ \begin{array}{l} 28 + 16 \times \textit{number-of-tables-inside-SQL} \times 2 \\ + 12 \times (1 + \sum_{j=1}^{b} \textit{number-of-RDAREAs-that-define-table-c} \times 2 \end{array} \right\}$$

(bytes)

a:

Number of SQL statements to be executed

b:

Number of tables inside the *c*-th SQL statement

(10) Statistical information related to SQL object execution (pcd)

- HiRDB/Single Server

Formula

$\textit{Statistics log size} = 388 \times \{(a + b + c + d + e + f) + g + h\}$

(Bytes)

- HiRDB/Parallel Server

Formula

$\textit{Statistics log size} = 388 \times \{(a + b + c + d + e + f) \times i + g + h\}$

(Bytes)

a:

Number of `OPEN` statements to be executed

b:

Number of `CLOSE` statements to be executed

c:

Number of `INSERT` statements to be executed

d:

Number of `DELETE` statements to be executed

e:

Number of `ASSIGN LIST` statements to be executed

f:

Number of `UPDATE` statements to be executed

g:

Number of `FETCH` statements to be executed

h:

Number of `DESCRIBE` statements to be executed

i:

Value obtained from the following formula:

$$\sum_{k=1}^{n} \left\{ 1 + \sum_{m=1}^{Tm} \textit{number-of-RDAREAs-that-define-table-m} \times 2 \right\}$$

n:

Number of SQL statements to be executed

Tm:

Number of tables inside the *m*-th SQL statement

(11) Statistical information related to SQL statement statistics (sqh)

Formula

Statistics log size = (728 + *a*) × *b* (Bytes)

a:

Average SQL length of SQL statements

b:

Number of SQLs to be executed during statistical information collection

If the total number of SQLs that will actually be executed is known, use the computation formula above. If the number of SQLs that will be executed during statistical information collection is not known, that number can be determined with the following formula:

Number of SQLs[*] = *average-number-of-SQL-statements-executed-per-UAP* × (*number-of-UAPs-to-be-executed-per-unit-time* × *(statistical-information-collection-period* ÷ *unit-time*)

[*] If a stored procedure or stored function will be executed, the SQL statements in the procedure or function must also be counted.

(12) Statistics on SQL object transmission (obj)

Formula

Statistics log size = 300 × *number-of-SQL-statements-to-be-executed-during-statistical-information-collection* (Bytes)

If it is possible to determine the cumulative number of SQL statements to be executed, this formula can be used to make that determination. If the number of SQL statements to be executed during statistical information collection is not known, the number of SQL statements can be determined using the following formula:

Number of SQL statements[*] =
average-number-of-SQL-statements-to-be-executed-in-UAP × *(number-of-UAP-executions-per-unit-of-time* × *statistical-information-collection-time* ÷ *unit-of-time)*

[*] If a stored procedure or stored function is to be executed, the SQL statements described in the procedure or function must also be included in the count.

(13) Statistical information related to foreign server operation (fsv)

Formula

Statistics log size = 188 × *a* × *b* (bytes)

a: Number of foreign servers for which foreign server information is defined

b: Number of transactions on which statistical information is being collected

(14) Statistical information related to foreign server usage status (hba)

Formula

Statistics log size = 404 × *a* (bytes)

a: Number of SQL statements for the foreign server to be executed during statistical information collection

Number of SQL statements =
average-number-of-SQL-statements-for-foreign-server-executed-by-UAP × (*UAP-execution-count-per-unit-time* × *statistical-information-collection-time* ÷ *unit-time*)

D.2 Formulas for determining size of SQL object buffer (pd_sql_object_cache_size)

For information on the variables used in the formulas, see (3) following.

(1) Formula for determining the size of the SQL object in an SQL statement

The following formula is used to determine the size of the SQL object in an SQL statement:

Formula

Size of the SQL object in an SQL statement (KB) =

$\uparrow\{$

$1840 + 46 \times RCN + 298 \times Si + 20 \times Pi + 1138 \times Ti + 76 \times Ti \times Di + 80 \times Ci + 40 \times Ii + 534 \times Wi + 20 \times Ki + Li + 8 \times TCi + 656 \times Di + 48 \times nFF + 100 \times nFP + 148 \times nFC + 696 \times nPFF + 16 \times (nAT + nPAT) + 20 \times nCAT + 28 \times (nAF + nCAF) + 20 \times (nAA + nPAA + nCAA) + 1057 \times nSPA + 120 \times nSPP + 287 \times nSFF + 8 \times nSFP + 813 \times nJFC + 20 \times nJFP$

$[+ 60 \times nFT + nFTS + 32 \times nFTQ]^{1}$

$[+ 1057 \times nTR + 120 \times (nTSN + nTSO) + 20 \times (nTCN + nTCO)]^{2}$

$[+ 760 + 376 \times RCC + 1880 \times RCT]^{3}$

$[+ 32 \times Si + 16]^{4}$

$[+ \uparrow(42 \times SiT) + \{52 + 152 \times (SiTA + SiSA + SiNA) \times (SiT + SiS + SiN)\} \uparrow]^{5}$

$\} \div 1024 \uparrow$

[1] Add the result of this formula when specifying a foreign table.

[2] Add the result of this formula when using a trigger.

[3] Add the result of this formula when using a referential constraint.

[4] Computation formula for column name description area size. Add the result of this formula when using dynamic SQL.

[5] Computation formula for type name description area size. Add the result of this formula when using dynamic SQL.

Notes:

- If you use stored procedures or stored functions, also include the SQL statements described in the procedures or functions in the computation.
- If you use triggers, also include the SQL statements described in the triggers in the computation.
- If you use stored procedures, stored functions, or triggers, compute the size of the routine control object separately for the stored procedures, stored functions, or triggers, and add the result to the size of the SQL object buffer. The formula for computing the size of a routine control object is shown in (2) following.

(2) Formula for determining the size of the routine control object of a routine

(a) When the user defines a trigger

When the user defines a stored procedure, a stored function, or a trigger, the following formula is used to determine the size of the routine control object of a routine:

Formula

Size of the routine control object of a routine (KB) =

$$\begin{gathered}\uparrow \{ \\ 600 + 28 \times sRi + 32 \times (sRUi + sDi) + 56 \times sSXi + sCUi + sSi + sPi + sLA + sKi + sL + 80 \times sWi + 24 \times sCM + 32 \times sCCR + 2 \times sDCR + 60 \times sCHD + 72 \times sDHD + 64 \times sHCN + 8 \times sCHD \times sHCN + 48 \times nRFF + 100 \times nRFP + 148 \times nRFC + 200 \times nPRFF + 8 \times nPRFP + 196 \times nPA + 64 \times nPP + 36 \times nPPI + 20 \times nPPO + 200 \times nPPA + 8 \times nPPP + 20 \times nAR + 48 \times nARA + 16 \times nRPAT + 20 \times nCAT + 28 \times (nRPAF + nRCAF) + 20 \times (nRPAA + nRCAA) + 287 \times nRSFF + 8 \times nRSFP + 813 \times nPJA + 20 \times nPJP + 813 \times nRJFC + 20 \times nRJFP \\ [+ 28 \times (nTSN \times 2 + nTSO)]^{*} \\ \} \div 1024 \uparrow \end{gathered}$$

[*] Add the result of this formula when using a trigger.

(b) When HiRDB automatically creates a trigger

When HiRDB creates a trigger for constraint control when `CASCADE` is specified for the referential action during table definition, the following formula is used to determine the size of the routine control object of a routine:

Formula

Size of the routine control object of a routine (KB) =

$$\begin{gathered}\uparrow \{ \\ 240 + 608 \times RCC + (5120 + 100 \times RDi + 256 \times RIi) \times RCP \times RCT \\ \} \div 1024 \uparrow \end{gathered}$$

(3) Variables used in the formulas

Variable name	Explanation
RCN	Total number of tables and indexes used by the SQL object
Si	Number of retrieval items inside the SQL statement (number of columns if the columns specified by the SQL statement are index columns)
Pi	Number of embedded variables or parameters inside the SQL statement
Ti	Number of table names inside the SQL statement
Ci	Number of column names inside the SQL statement
TCi	Number of table columns inside the SQL statement
Wi	Number of boolean operators inside the SQL statement[1]
Ki	Number of constants inside the SQL statement[1]

Variable name	Explanation
Li	Total size of the constants inside the SQL statement[1](units: bytes)
Ii	Number of indexes used during SQL statement execution (number of indexes specified as retrieval conditions in the table specified by the SQL statement)
Di	Total number of storage conditions defined in the table inside the SQL statement (times 2 for a matrix-partitioned table)
SiT	Number of abstract data types in the selection expression inside the SQL statement
SiS	Number of supertypes for the abstract data types in the selection expression inside the SQL statement
SiN	Total number of supertypes for the abstract data type, which is a subtype of the selection expression inside the SQL statement
SiTA	Number of attributes for the abstract data types in the selection expression inside the SQL statement
SiSA	Number of attributes of the supertypes for the abstract data types in the selection expression inside the SQL statement
SiNA	Total number of components specified for the abstract data type, which is a subtype of the selection expression inside the SQL statement
nSPA	Number of times the procedure statements inside the SQL statement are invoked
nSPP	Total number of arguments for the procedure statements inside the SQL statement
nFF	Number of times the functions inside the SQL statement are invoked[1]
nFP	Total number of function arguments inside the SQL statement[1]
nFC	Total number of function definition candidates for the functions inside the SQL statement (number of function invocations *nFF* + number of function definitions that use as arguments the subtypes to the function invocations whose arguments are abstract data types)
nPFF	Number of times the plug-in functions used by SQL objects are invoked (1 for the number of plug-in function invocations inside the SQL statement + `SELECT`; 6 for `INSERT`, `UPDATE`, or `DELETE`)
nSFF	Number of times the system definition scalar functions inside the SQL statement are invoked[1]
nSFP	Total number of arguments for the system definition scalar functions inside the SQL statement[1]
nJFC	Number of times the Java functions inside the SQL statement are invoked
nJFP	Total number of arguments of the Java functions inside the SQL statement

Variable name	Explanation
nAT	Number of abstract data types used by component specification inside the SQL statement (excluding the abstract data types that are manifested by supertypes and abstract data type attributes)
nAA	Number of abstract data types used by component specification inside the SQL statement (including the abstract data types that are manifested by supertypes and abstract data type attributes)
nAF	Total number of attributes used by component specification inside the SQL statement
nPAT	Number of abstract data types used by the arguments of the plug-in functions used by the SQL object (excluding the abstract data types that are manifested by supertypes and abstract data type attributes)
nPAA	Number of abstract data types used by the arguments of the plug-in functions used by the SQL object (including supertypes and subtypes)
nCAT	Number of times the constructor functions inside the SQL statement are invoked
nCAA	Number of abstract data types of the constructor functions inside the SQL statement (including supertypes)
nCAF	Total number of attributes of the abstract data types of the constructor functions inside the SQL statement
nFT	Number of foreign tables inside the SQL statement
nFTS	Size of the SQL statement that retrieves foreign tables[2]
nFTQ	Number of embedded variables or parameters inside the SQL statement that retrieves foreign tables[2]
nTR	Number of triggers that are activated by SQL statement execution
nTSN	Total number of columns modified by the new values correlation name inside the trigger SQL statement of each trigger activated by SQL statement execution
nTSO	Total number of columns modified by the old correlation name inside the trigger SQL statement of each trigger activated by SQL statement execution
nTCN	Total number of columns modified by the new values correlation name inside the trigger action condition of each trigger activated by SQL statement execution
nTCO	Total number of columns modified by the old correlation name inside the trigger action condition of each trigger activated by SQL statement execution
RCC	Total number of member columns of the foreign key and the primary key of the table that references the update-target table inside the SQL statement

Variable name	Explanation
RCT	Total number of tables that reference the update-target table and the number of tables that the update-target table references inside the SQL statement
RCP	Total number of cascades specified for referencing actions during referencing table definition
RIi	Total number of indexes defined for referenced tables that are specified for referencing during referencing table definition
RDi	Total number of partitioning storage conditions defined for referenced tables that are specified for referencing during referencing table definition (to be multiplied by 2 for a matrix partitioning table)
sRi	Number of SQL parameters inside procedures and functions (Count for SQL parameter with `INOUT` specification should be multiplied by 2).
sRUi	Total number of SQL parameters inside procedures and functions (or the total number of columns modified by an old or new values correlation name inside the trigger SQL statement inside trigger definition)
sDi	Number of SQL variables (`declare`) inside procedures, functions, and trigger SQL statements
sSXi	Total number of SQLCODE and SQLCOUNT variables inside procedures, functions, and trigger SQL statements
sCUi	Total number of `CURRENT_TIME` and `CURRENT_DATE` constants inside procedures, functions, and trigger SQL statements
sSi	Number of data manipulation SQL statements inside procedures and trigger SQL statements (SQL statements excluding cursor declaration: `OPEN`, `FETCH`, `CLOSE`, `UPDATE`, `DELETE`, and `INSERT` statements, for example)
sPi	Number of routine control SQL statements inside procedures, functions, and trigger SQL statements (`BEGIN`, `SET`, `IF`, `ELSE IF`, and `WHILE`, for example)
sLA	Number of labels inside procedures, functions, and trigger SQL statements
sKi	Number of constants inside procedures, functions, and trigger SQL statements (excluding the constants of data manipulation SQL statements described inside procedures and trigger SQL statements)
sL	Combined total size of the constants inside procedures, functions, and trigger SQL statements (excluding the constants of data manipulation SQL statements described inside procedures and trigger SQL statements)
sWi	Number of condition predicates inside procedures, functions, and trigger SQL statements
sCM	Number of compound statements inside procedures, functions, and trigger SQL statements
sCCR	Number of compound statements that describe cursor declarations for procedures and trigger SQL statements

Variable name	Explanation
sDCR	Number of cursor declarations for procedures and trigger SQL statements
sCHD	Number of compound statements that describe handler declarations for procedures, functions, and trigger SQL statements
sDHD	Number of handler declarations for procedures, functions, and trigger SQL statements
sHCN	Number of condition values described inside handler declarations for procedures, functions, and trigger SQL statements
nRFF	Number of function invocations inside the routine
nRFP	Total number of function arguments inside the routine
nRFC	Total number of function definition candidates for the functions inside the routine (number of function invocations *nFF* + number of function definitions that use as arguments the subtypes to the function invocations whose arguments are abstract data types)
nPRFF	Number of times the plug-in functions used by SQL objects of the routine are invoked
nPRFP	Total number of plug-in parameters of the plug-in functions used by the SQL objects of the routine
nPA	Number of procedure invocations inside the routine
nPP	Total number of parameters for the procedures inside the routine
nPPI	Total number of input parameters for the procedures inside the routine (including input/output parameters)
nPPO	Total number of output parameters for the procedures inside the routine (including input/output parameters)
nPPA	Number of times the plug-in procedure used by the SQL objects of the routine is invoked
nPPP	Total number of plug-in parameters of the plug-in procedure used by the SQL objects of the routine
nRSFF	Number of times the system definition scalar functions inside the routine are invoked
nRSFP	Total number of arguments for the system definition scalar functions inside the routine
nPJA	Number of times the Java procedures inside the routine are invoked
nPJP	Total number of arguments for the Java procedures inside the routine
nRJFC	Number of times the Java functions inside the routine are invoked
nRJFP	Total number of arguments for the Java functions inside the routine
nAR	Number of abstract data types used by component specification inside the routine (excluding the abstract data types that are manifested by supertypes and abstract data type attributes)

Variable name	Explanation
nARA	Total number of attributes used in component specification inside the routine
nRPAT	Number of abstract data types used by the parameters of the plug-in routines used by the SQL objects of the routine (excluding the abstract data types of supertypes and abstract data type attributes)
nRPAA	Number of abstract data types used by the parameters of the plug-in routines used by the SQL objects of the routine (including supertypes)
nRPAF	Total number of attributes of the abstract data types used by the parameters of the plug-in routines used by the SQL objects of the routine
nRCAT	Number of times the constructor functions inside the routine are invoked
nRCAA	Number of abstract data types of the constructor functions inside the routine (including supertypes)
nRCAF	Total number of attributes of the abstract data types of the constructor functions inside the routine

[1] When triggers are used, the trigger action conditions for the individual triggers that are activated by SQL statement execution must also be counted.

[2] You can use the access path display utility (pdvwopt command) to check for the SQL statements that retrieve foreign tables.

D.3 Formulas for determining size of table definition information buffer (pd_table_def_cache_size)

(1) Variables used in the formulas

a: Value of DEFINITION_CACHE_SIZE (units: bytes) of a dictionary table (SQL_TABLES table)

If you do not know the value of DEFINITION_CACHE_SIZE, see *(3) Determining the table definition cache size*.

b: Number of columns in the table

Substitute 8 for a reflection status management table.

c: Number of indexes in the table

Substitute 1 for a reflection status management table.

d: Number of table-partitioning conditions

For a reflection status management table, substitute the number of servers to which the RDAREAs specified in the -r and -o options of the pdorcreate command belong.

e: Number of RDAREAs for table

For a reflection status management table, substitute the number of RDAREAs specified in the `-r` option of the `pdorcreate` command + 1.

f: Number of RDAREAs for index

For a reflection status management table, substitute the number of RDAREAs specified in the `-r` option of the `pdorcreate` command + 1.

g: Number of BLOB columns

h: Number of abstract data types

i: Number of plug-in options

j: Number of plug-in indexes

k: Total number of applicable functions

m: Total number of BLOB attributes in abstract data types

n: Number of index exceptional key values

p: Value of `STATISTICS_CACHE_SIZE` (units: bytes) of a dictionary table (`SQL_TABLES` table)

Add this value when you use the `pdgetcst` command to obtain table optimization information. Because the result is in bytes, it must be converted into kilobytes before being entered in the formulas. If you do not know the value of `STATISTICS_CACHE_SIZE`, use the following formula to determine it:

$2.6 \times q + 0.04 \times c + 0.02$ (units: KB)

q: Number of column optimization information items

r: Number of abstract data types containing BLOB attributes

s: Number of columns specified by the `DEFAULT` operand of `CREATE TABLE`

t: Total size of the default values to be specified by the `DEFAULT` operand of `CREATE TABLE`

Add the sizes of all columns for which default values are specified. If the default values may be increased, take the increases into consideration.

u: Number of authorization identifiers that have the same table identifier as the table name of the public view table

(2) Table definition information buffer size per table (KB)

To determine the table definition information buffer size per table, use the following approximation formulas. Note that for a view table, also determine the size of the base table or foreign table that becomes the base for the view table.

Table type	Formula for determining the table definition information buffer size per table (units: KB)
Table definition information buffer size per base table or foreign table	$\uparrow\{(4+\uparrow a \div 1024 \uparrow + 0.01 \times b + p + 7) \div 8\}\uparrow \times 8$
Table definition information buffer size per view table	$\uparrow\{(4+\uparrow a \div 1024 \uparrow + 0.01 \times b + 7) \div 8\}\uparrow \times 8$
Table definition information buffer size for reflection status management table (HiRDB Staticizer Option)	$\uparrow\{(4+\uparrow a \div 1024 \uparrow + 0.01 \times 8 + 7) \div 8\}\uparrow \times 8$

(3) Determining the table definition cache size

(a) When you do not know the value of DEFINITION_CACHE_SIZE

If any of the following conditions is applicable, the value of `DEFINITION_CACHE_SIZE` cannot be determined, even if a dictionary table is retrieved:

- Table has not yet been defined
- Version was upgraded from HiRDB Version 4.0 (in this case, the value of `DEFINITION_CACHE_SIZE` is not correct)[1]
- Table was converted from 32-bit-mode HiRDB to 64-bit-mode HiRDB[1, 2]

[1] In this case, the value in DEFINITION_CACHE_SIZE is not the correct value.

[2] Whether or not a table has been converted from 32-bit-mode HiRDB to 64-bit-mode HiRDB can be determined by looking at the time of execution of the `pdvrup` command and the time of table creation. If the table's creation time is earlier, the table has been converted from the 32-bit mode to the 64-bit mode. The `pdvrup` command's execution time can be determined from CHANGE_TIME of MASTER.SQL_TABLES in the SQL_TABLES dictionary table. A table's creation time can be determined from CREATE_TIME for the created table in the SQL_TABLES dictionary table. These columns can be retrieved with the following SQL statements:

- `pdvrup` command execution time

```
select CHANGE_TIME from MASTER.SQL_TABLES
  where TABLE_SCHEMA='MASTER' andTABLE_NAME='SQL_TABLES'
```

- Table creation time

```
select CREATE_TIME from MASTER.SQL_TABLES
  where TABLE_SCHEMA='authorization identifier' and
TABLE_NAME='table-name'
```

(b) Formula for determining the table definition cache size

Use the following formulas to determine the table definition cache size.

- Base table or foreign table: Formula 1 + Formula 2
- View table: Formula 1
- Reflection status management table: Formula 1

Formula type	Formula (units: KB)
Formula 1	■ For 32-bit mode HiRDB $0.6 + 0.1 \times b + 0.13 \times c + 0.13 \times (d + 1) + 0.01 \times (e + f)$ ■ For 64-bit HiRDB $0.9 + 0.13 \times b + 0.16 \times c + 0.13 \times (d + 1) + 0.01 \times (e + f)$
Formula 2	■ Add the following if a LOB column is defined for the table: $(0.01 + 0.02 \times e) \times g$ ■ Add the following if an abstract data type is defined for the table: $+ 0.3 \times h + 0.3 \times i$ ■ Add the following if a plug-in index is defined for the table: $+ 0.52 \times j + 0.01 \times k$ ■ Add the following if an abstract data type containing a LOB attribute is defined for the table: $+ 0.01 \times r + (0.02 + 0.01 \times e) \times m$ ■ Add the following if an index exclusion key value is defined for the table: $+ 0.3 \times n$ ■ Add the following if `SEGMENT REUSE` is specified for the table (excludes `SEGMENT REUSE NO`): $+ 0.01 \times e$ ■ Add the following if a default value is specified for the DEFAULT operand in the table: $+ 0.01 \times s + t$ ■ Add the result of the following formula when using a public view table: $+ 8 + (u \times 32)$

D.4 Formula for determining total number of tables and RDAREAs per server locked with UNTIL DISCONNECT specified (pd_lck_until_disconnect_cnt)

The following formula is used to determine the total number of tables and RDAREAs per server locked with `UNTIL DISCONNECT` specified (`pd_lck_until_disconnect_cnt`). Note that the formula differs depending on the server type.

Formula

Server type	Formula
Single server	$(a + b + c + d + o + p + q) \times 2 + \{2 \times (f + g + i + 1) + (g \times h)\} \times e$
Dictionary server	$(a + b + c + d) \times 2 + \{2 \times (f + g + i + 1) + (g \times h)\} \times e$
Back-end server	$(a + b + c + d + o + p + q + r + s) \times 2 + \{2 \times (f + g + i + 1) + (g \times h)\} \times e$

a:

Number of tables specified in `LOCK TABLE` statements with `UNTIL DISCONNECT` specified that are to be concurrently executed

b:

Number of RDAREAs that store the tables specified in `LOCK TABLE` statements with `UNTIL DISCONNECT` specified that are to be concurrently executed

c: 1

This is the number for inner replica configuration management. Add this value when using the inner replica facility.

d:

Number of RDAREAs that store the tables specified in `LOCK TABLE` statements with `UNTIL DISCONNECT` specified that are to be concurrently executed

This is the number for replica group configuration management. Add this value when using the inner replica facility.

e:

Number of utilities that are concurrently executed

f:

Number of LOB RDAREAs that store the tables to be processed by the utilities that are concurrently executed

g:

Number of RDAREAs that store the indexes to be processed by the utilities that are concurrently executed

h:

Number of indexes to be processed by the utilities that are concurrently executed

i:

Number of RDAREAs that store the tables to be processed by the utilities that are concurrently executed.

o:

Number of RDAREAs that store the tables to which the local buffers for data that are to be used concurrently are allocated

Add this number when using local buffers for data.

p:

Number of RDAREAs that store the tables to which the local buffers for index that are to be used concurrently are allocated

Add this number when using local buffers for index.

q:

Total number of tables that are the target of the indexes to which the local buffers for index that are to be used concurrently are allocated

Add this number when using local buffers for index.

r:

Total number of shared table storage RDAREAs specified by `LOCK` statements with `IN EXCLUSIVE MODE` specified that are to be concurrently executed

Add this number when using shared RDAREAs.

s:

Total number of all shared RDAREAs for index defined in the shared tables that are specified by `LOCK` statements with `IN EXCLUSIVE MODE` specified that are to be concurrently executed

Add this number when using shared RDAREAs.

Note: Utilities in this case includes the following:

- Database load utility
- Database reorganization utility
- Free page release utility
- Global buffer residence utility
- Rebalancing utility (exclusive mode)

D.5 Formulas for determining size of routine definition information buffer (pd_routine_def_cache_size)

The following formula is used to determine the definition information size for routines:

total size of definition information for routines that are used frequently
+ total size of definition information for plug-in functions of plug-ins used
+ total size of definition information for system definition scalar functions that are used frequently

(1) Determination of routine definition information size per routine

The following formula is used to determine one routine's definition information size:

Formula

$\uparrow(1.3 + 0.2 \times a)\uparrow \times b$ (KB)

a: Total number of parameters for routines that are used frequently

b: Number of definitions for routines that are used frequently

(2) Determination of definition information size of a plug-in function

The formula provided below is used to determine the definition information size of a plug-in function:

Formula

$0.6 + c + 0.2 \times d$ (KB)

c:

Total number of plug-in functions used in DML by a single plug-in*

d:

Total number of parameters of plug-in functions used in DML by a single plug-in*

Note

This formula is for one plug-in. If multiple plug-ins are installed, the total for all of the installed plug-ins should be used.

* The SQL shown below should be used to determine the total number of plug-in functions used in DML and the parameters of the plug-in functions used in DML:

```
SELECT COUNT(*),SUM(N_PARAM) FROM
MASTER.SQL_PLUGIN_ROUTINES
  WHERE PLUGIN_NAME = 'plug-in-name'
  AND (TIMING_DESCRIPTOR = 'ADT_FUNCTION'
    OR TIMING_DESCRIPTOR IS NULL
    OR TIMING_DESCRIPTOR = 'BEFORE_INSERT'
    OR TIMING_DESCRIPTOR = 'AFTER_INSERT'
    OR TIMING_DESCRIPTOR = 'BEFORE_UPDATE'
OR TIMING_DESCRIPTOR = 'AFTER_UPDATE'
OR TIMING_DESCRIPTOR = 'BEFORE_DELETE'
OR TIMING_DESCRIPTOR = 'AFTER_DELETE'
OR TIMING_DESCRIPTOR = 'BEFORE_PURGE_TABLE'
OR TIMING_DESCRIPTOR = 'AFTER_PURGE_TABLE'
OR TIMING_DESCRIPTOR = 'INDEX_SEARCH'
OR TIMING_DESCRIPTOR = 'INDEX_COUNT'
OR TIMING_DESCRIPTOR = 'INDEX_INSERT'
OR TIMING_DESCRIPTOR = 'INDEX_BEFORE_UPDATE'
OR TIMING_DESCRIPTOR = 'INDEX_AFTER_UPDATE'
OR TIMING_DESCRIPTOR = 'INDEX_DELETE'
OR TIMING_DESCRIPTOR = 'PURGE_INDEX'
OR TIMING_DESCRIPTOR = 'INDEX_MAINTENANCE_DEFERRED'
OR TIMING_DESCRIPTOR = 'BEFORE_INSERT_DC'
OR TIMING_DESCRIPTOR = 'BEFORE_UPDATE_DC'
OR TIMING_DESCRIPTOR = 'BEFORE_DATA_CHECK'
OR TIMING_DESCRIPTOR = 'AFTER_DATA_CHECK')
```

(3) Determination of each definition scalar function's definition information size for each function

Table D-1 is used to determine the definition information size for each system definition scalar function:

Table D-1: Definition information size for each system definition scalar function

Functions	Definition information size (KB)
ACOS, ADD_INTERVAL, ASCII, ASIN, ATAN, ATAN2, CENTURY, COS, COSH, CHR, DATE_TIME, DAYNAME, DAYOFWEEK, DAYOFYEAR, DEGREES, EXP, INTERVAL_DATETIMES, LAST_DAY, LN, LOG10, MIDNIGHTSECONDS, MONTHNAME, MONTHS_BETWEEN, NEXT_DAY, PI, RADIANS, SIN, SINH, SQRT, TAN, TANH, WEEK, WEEKOFMONTH, YEARS_BETWEEN	2
POWER, IS_DBLBYTES, IS_SNGLBYTES, ISDIGITS, ROUNDMONTH, TRANSL_LONG	4
CEIL, FLOOR, HALF, INSERTSTR, INSERTSTR_LONG, LEFTSTR, LTRIMSTR, NUMEDIT, QUARTER, REPLACE_LONG, REVERSESTR, RIGHTSTR, RTRIMSTR, SIGN, STRTONUM, TRUNCYEAR	6
LTRIM, REPLACE, RTRIM, TRANSL, TRUNC	12
POSSTR, ROUND	18
GREATEST, LEAST	32

E. Determining the Number of Locked Resources

This appendix provides and explains formulas for determining the number of locked resources needed in order to execute the various SQL statements. The maximum number of locked resources is determined, some buffer space is added, and a pool size is set in the following operands:

- `pd_lck_pool_size` operand in each server's server definition
- `pd_fes_lck_pool_size` operand in the front-end server definition
- `pd_until_disconnect_cnt` operand in the server definition of each server

Notes

1. The number of locked resources count determined here is valid within a transaction. When multiple SQLs are executed in a single transaction, the total of their locked resources is needed. However, it is not needed if locking has already been applied. For details about locking, see the *HiRDB Version 8 UAP Development Guide*.
2. For a HiRDB/Parallel Server, the number of resources for each back-end server must be computed in terms of the number of resources (RDAREAs, indexes, lines) managed by the applicable back-end server.
3. The number of locked resources that is added when the `pd_inner_replica_control` operand is specified is requested only once during transaction execution.

E.1 Definition SQLs

(1) CREATE ALIAS

Condition	Number of lock requests
HiRDB/Single Server	13
HiRDB/Parallel Server (Dictionary server)	13

(2) CREATE CONNECTION SECURITY

Condition	Number of lock requests
HiRDB/Single Server	7 + *number-of-users-registered-in-dictionary-table-SQL_USERS*

Condition		Number of lock requests
HiRDB/Parallel Server	Front-end server	1
	Dictionary server	6 + *number-of-users-registered-in-dictionary-table-SQL_USERS*

(3) CREATE FOREIGN INDEX

Condition	Number of lock requests
Front-end server	1
Dictionary server	20 + *number of member columns* × 5 + 2

(4) CREATE FOREIGN TABLE

Condition	Number of lock requests
Dictionary server	17 + *column count* × 4 + 2

(5) CREATE FUNCTION

Condition		Number of lock requests
HiRDB/Single Server		*number of locked resources in* `CREATE PROCEDURE` Add the following when parameters are involved: + *parameters count* × 2 Add the following when a parameter includes a user-defined type: + *user-defined types count* × 2
HiRDB/Parallel Server	Front-end server and Back-end server	*same as* `CREATE PROCEDURE`
	Dictionary server	*number of locked resources in* `CREATE PROCEDURE` Add the following when parameters are involved: + *parameters count* × 2 Add the following when a parameter includes a user-defined type: + *user-defined types count* × 2

(6) CREATE INDEX (not a plug-in index)

Condition		Number of lock requests
HiRDB/Single Server*		25 + *number of routines in which indexes become invalid* + *number of RDAREAs for index* × 6 + *number of member columns* × 4 Add the following if there are routines that reference a table that defines an index: + *number of routines that reference the table that defines an index* × 4 + 2 Add the following if there are trigger action procedures that reference a table that defines an index: + *number of trigger action procedures that reference the table that defines an index* × 5 Add the following if there are two or more RDAREAs for an index: + *number of RDAREAs for index* × 3 + 1 Add the following if the inner replica facility is used and a LOB column or LOB attribute is defined in the table for which indexes are defined: + *number of RDAREAs for LOB column* + *number of RDAREAs for LOB attribute*
HiRDB/Parallel Server	Front-end server	1 + *number of routines in which indexes become invalid*
	Dictionary server	19 + *number of RDAREAs for index* × 4 + *number of member columns* × 4 Add the following if there are routines that reference a table that defines an index: + *number of routines that reference the table that defines an index* × 4 + 2 Add the following if there are trigger action procedures that reference a table that defines an index: + *number of trigger action procedures that reference the table that defines an index* × 5 Add the following if there are two or more RDAREAs for an index: + *number of RDAREAs for index* × 3 + 1 + *number of routines in which indexes become invalid*
	Back-end server*	5 + *number of RDAREAs for index* × 2 + *number of routines in which indexes become invalid* Add the following if the inner replica facility is used and a LOB column or LOB attribute is defined in the table for which indexes are defined: + *number of RDAREAs for LOB column* + *number of RDAREAs for LOB attribute*
	Reference-only back-end server*	2

* If the inner replica facility is used, a number of lock requests equaling the number of

generations is required.

(7) CREATE INDEX (plug-in index)

Condition		Number of lock requests
HiRDB/Single Server*		21 + *number of routines in which indexes become invalid* + *number of RDAREAs for index* × 8 + *number of member columns* × 4 + *number of functions applied* × 5 + Σ *number of parameters of the functions applied* Add the following if the inner replica facility is used and a LOB column or LOB attribute is defined in the table for which indexes are defined: + *number of RDAREAs for LOB column* + *number of RDAREAs for LOB attribute*
HiRDB/Parallel Server	Front-end server	1 + *number of routines in which indexes become invalid*
	Dictionary server	15 + *number of RDAREAs for index* × 6 + *number of member columns* × 4 + *number of functions applied* × 5 + Σ *number of parameters of the functions applied* + *number of routines in which indexes become invalid*
	Back-end server*	5 + *number of RDAREAs for index* × 2 + *number of routines in which indexes become invalid* Add the following if the inner replica facility is used and a LOB column or LOB attribute is defined in the table for which indexes are defined: + *number of RDAREAs for LOB column* + *number of RDAREAs for LOB attribute*

* If the inner replica facility is used, a number of lock requests equaling the number of generations is required.

(8) CREATE PROCEDURE

Condition	Number of lock requests
HiRDB/Single Server	13 + *number of tables accessed in SQL statement preprocessing* + *number of view tables accessed in SQL statement preprocessing* + *number of base tables that are the basis of view tables accessed in SQL statement preprocessing* Add the following if a resource to be used exists: + *number of resources to be used* × 7 + 3 Add the following if parameters are specified: + *number of parameters* × 3 + 1 Add the following if the parameter data type is a user-defined type: + *number of user-defined type parameters* × 2 + 1

Condition		Number of lock requests
HiRDB/Parallel Server	Front-end server	3 + *number of tables accessed in SQL statement preprocessing* + *number of view tables accessed in SELECT statement preprocessing* + *number of base tables that are the basis of view tables accessed in SELECT statement preprocessing*
	Dictionary server	15 Add the following if a resource to be used exists: + *number of resources to be used* × 7 + 3 Add the following if parameters are specified: + *number of parameters* × 3 + 1 Add the following if the parameter data type is a user-defined type: + *number of user-defined type parameters* × 2 + 1

(9) CREATE SCHEMA

Condition	Number of lock requests
HiRDB/Single Server	5
HiRDB/Parallel Server (Dictionary server)	

(10) CREATE SERVER

Condition	Number of lock requests
Dictionary server	9 + `pd_max_foreign_server` operand value

(11) CREATE TABLE

Condition		Number of lock requests
HiRDB/Single Server*	RDAREA for table is omitted and the table is not row-partitioned	29 + *number of public user RDAREAs* × 3 + *number of columns* × 3 + *number of LOB columns* × 4 Add the following if a user-defined type is used: + *number of user-defined types* × 6 + *number of LOB attributes* × *number of partitioned RDAREAs* × 4 + *number of user-defined attributes* × 2 + 5 Add the following if a cluster key is defined: + 5 + *number of RDAREAs for index* × 2 + *number of partitioned RDAREAs* × 6 + 8 + *number of member columns* × 3 Add the following if a referential constraint is defined: + *number of referenced tables* + *number of foreign keys* × 10 Add the following if CASCADE is defined for referential constraint operation: +*number of CASCADEs* × 17 +*number of referenced tables specified by CASCADE* × 4 Add the following if the specification of referential constraint operation is UPDATE ON CASCADE: + *total number of primary key member columns of the referenced table for which CASCADE is specified* × 8 + *number of referenced tables specified by CASCADE* × 3 Add the following if there are functions that become invalid: + *number of functions whose objects are invalid* + Σ (2 + *number of resources inside procedure* × 5) + *total number of functions that reference the referenced tables* Add the following if a check constraint is defined: + *number of check constraints* × 9

Condition		Number of lock requests
	RDAREA for table is specified and the table is not row-partitioned	28 + *number of columns* × 4 + *number of LOB columns* × 4 + *number of RDAREAs for table* × 4 Add the following if a user-defined type is used: + *number of user-defined types* × 6 + *number of LOB attributes* × *number of partitioned RDAREAs* × 4 + *number of user-defined attributes* × 2 + 5 Add the following if a cluster key is defined: + 5 + *number of RDAREAs for index* × 2 + *number of partitioned RDAREAs* × 6 + 8 + *number of member columns* × 3 Add the following if a referential constraint is defined: + *number of referenced tables* + *number of foreign keys* × 10 Add the following if CASCADE is defined for referential constraint operation: +*number of CASCADEs* × 17 +*number of referenced tables specified by CASCADE* × 4 Add the following if the specification of referential constraint operation is UPDATE ON CASCADE: + *total number of primary key member columns of the referenced table for which CASCADE is specified* × 8 + *number of referenced tables specified by CASCADE* × 3 Add the following if there are functions that become invalid: + *number of functions whose objects are invalid* + Σ (2 + *number of resources inside procedure* × 5) + *total number of functions that reference the referenced tables* Add the following if a check constraint is defined: + *number of check constraints* × 9

Condition		Number of lock requests
	Row-partitioned table	29 + *number of columns* × 4 + *number of partitioned RDAREAs* × 21 + *number of LOB columns* × *number of partitioned RDAREAs* × 4 Add the following if a user-defined type is used: + *number of user-defined types* × 6 + *number of LOB attributes* × *number of partitioned RDAREAs* × 4 + *number of user-defined attributes* × 2 + 5 Add the following if a cluster key is defined: + 5 + *number of RDAREAs for index* × 2 + *number of partitioned RDAREAs* × 6 + 8 + *number of member columns* × 3 Add the following if a referential constraint is defined: + *number of referenced tables* + *number of foreign keys* × 10 Add the following if CASCADE is defined for referential constraint operation: +*number of CASCADEs* × 17 +*number of referenced tables specified by CASCADE* × 4 Add the following if the specification of referential constraint operation is UPDATE ON CASCADE: + *total number of primary key member columns of the referenced table for which CASCADE is specified* × 8 + *number of referenced tables specified by CASCADE* × 3 Add the following if there are functions that become invalid: + *number of functions whose objects are invalid* + Σ (2 + *number of resources inside procedure* × 5) + *total number of functions that reference the referenced tables* Add the following if a check constraint is defined: + *number of check constraints* × 9
HiRDB/Parallel Server (front-end server)		3 Add the following if a referential constraint is defined: + *number of referenced tables* + *total number of routines that reference the referenced tables*

Condition		Number of lock requests
HiRDB/Parallel Server (Dictionary server)	RDAREA for table is omitted and the table is not row-partitioned	20 + *number of public user RDAREAs* × 3 + *number of columns* × 3 + *number of LOB columns* × *number of partitioned RDAREAs* × 4 Add the following if a user-defined type is used: + *number of user-defined types* × 6 + *number of LOB attributes* × *number of partitioned RDAREAs* × 4 + *number of user-defined attributes* × 2 + 5 Add the following if a cluster key is defined: + *number of partitioned RDAREAs* × 6 + 8 + *number of member columns* × 3 Add the following if a referential constraint is defined: + *number of foreign keys* × 10 Add the following if CASCADE is defined for referential constraint operation: +*number of CASCADEs* × 17 +*number of referenced tables specified by CASCADE* × 4 Add the following if the specification of referential constraint operation is UPDATE ON CASCADE: + *total number of primary key member columns of the referenced table for which CASCADE is specified* × 8 + *number of referenced tables specified by CASCADE* × 3 Add the following if there are functions that become invalid: + *number of functions whose objects are invalid* + Σ (2 + *number of resources inside procedure* × 5) Add the following if a check constraint is defined: + *number of check constraints* × 9

Condition		Number of lock requests
	RDAREA for table is specified and the table is not row-partitioned	25 + *number of columns* × 4 + *number of LOB columns* × 4 Add the following if a user-defined type is used: + *number of user-defined types* × 6 + *number of LOB attributes* × *number of partitioned RDAREAs* × 4 + *number of user-defined attributes* × 2 + 5 Add the following if a cluster key is defined: + *number of partitioned RDAREAs* × 6 + 8 + *number of member columns* × 3 Add the following if a referential constraint is defined: + *number of foreign keys* × 10 Add the following if CASCADE is defined for referential constraint operation: +*number of CASCADEs* × 17 +*number of referenced tables specified by CASCADE* × 4 Add the following if the specification of referential constraint operation is UPDATE ON CASCADE: + *total number of primary key member columns of the referenced table for which CASCADE is specified* × 8 + *number of referenced tables specified by CASCADE* × 3 Add the following if there are functions that become invalid: + *number of functions whose objects are invalid* + Σ (2 + *number of resources inside procedure* × 5) Add the following if a check constraint is defined: + *number of check constraints* × 9

Condition		Number of lock requests
	Row-partitioned table	Add the following when the table is not a matrix-partitioned table: 24 + *number of columns* × 4 + *number of partitioned RDAREAs* × 17 + *number of LOB columns* × *number of partitioned RDAREAs* × 4 Add the following when the table is a matrix-partitioned table: 21 + *number of columns* × 4 + *number of partitioned RDAREAs* (including overlapped ones) × 16 + *number of LOB columns* × *number of partitioned RDAREAs* × 4 + *number of partitioning keys* × 2 + (*number of storage conditions* + 2) × 2 + 2 Add the following if a user-defined type is used: + *number of user-defined types* × 6 + *number of LOB attributes* × *number of partitioned RDAREAs* × 4 + *number of user-defined attributes* × 2 + 5 Add the following if a cluster key is defined: + *number of partitioned RDAREAs* × 6 + 8 + *number of member columns* × 3 Add the following if a referential constraint is defined: + *number of foreign keys* × 10 Add the following if CASCADE is defined for referential constraint operation: +*number of CASCADEs* × 17 +*number of referenced tables specified by CASCADE* × 4 Add the following if the specification of referential constraint operation is UPDATE ON CASCADE: + *total number of primary key member columns of the referenced table for which CASCADE is specified* × 8 + *number of referenced tables specified by CASCADE* × 3 Add the following if there are functions that become invalid: + *number of functions whose objects are invalid* + Σ (2 + *number of resources inside procedure* × 5) Add the following if a check constraint is defined: + *number of check constraints* × 9
HiRDB/Parallel Server (Back-end server)*		5 + *number of RDAREAs for table* × 4 Add the following if a cluster key is defined: + 5 + *number of RDAREAs for index* × 2
HiRDB/Parallel Server (reference-only back-end server)*		2 Add the following if a cluster key is defined: + 1

* If the inner replica facility is used, a number of lock requests equaling the number of generations is required.

(12) CREATE TRIGGER

Condition		Number of lock requests
HiRDB/Single Server		21 + *number of routines in which objects become invalid* × 5 + *number of resources used by trigger action procedures* × 5 Add the following if a trigger action condition is specified: + *number of resources used inside trigger action condition specification* × 5 + 3 Add the following if a column name modified by an old or new values correlation name is used inside a trigger action procedure: + *number of column types modified by new values correlation names* × 3 + *number of column types modified by old values correlation names* × 3 + 3 Add the following if trigger event columns are specified: + *number of trigger event columns* × 3 Add the following if there are routines that become invalid: + *number of invalid routines* × 5 + *number of resources used by routines* × 5 Add the following if there are trigger action procedures in the routines that become invalid: + *number of trigger action procedures* × 5
HiRDB/Parallel Server	Front-end server	3 + *number of routines in which objects become invalid*
	Dictionary server	18 + *number of routines in which objects become invalid* + *number of resources used by trigger action procedures* × 5 Add the following if a trigger action condition is specified: + *number of resources used inside trigger action condition specification* × 5 + 3 Add the following if a column name modified by an old or new values correlation name is used inside a trigger action procedure: + *number of column types modified by new values correlation names* × 3 + *number of column types modified by old values correlation names* × 3 + 3 Add the following if trigger event columns are specified: + *number of trigger event columns* × 3 Add the following if there are routines that become invalid: + *number of invalid routines* × 5 + *number of resources used by routines* × 5 Add the following if there are trigger action procedures in the routines that become invalid: + *number of trigger action procedures* × 5
	Back-end server	*number of routines in which objects become invalid*

(13) CREATE TYPE

Condition		Number of lock requests
HiRDB/Single Server		22 + *number of routines that become invalid* + *number of attributes* × 3 + (*number of user-defined types* × 6 + 2)[1] + 11[2] + (*number of routines that become invalid* × 4 + 2 + *number of resources of the routines that become invalid* × 5 + 1)[3] + *number of locked resources for function/procedure definitions*[4]
HiRDB/Parallel Server	Front-end server	1 + *number of routines that become invalid* + *number of locked resources for function/procedure definitions*[4]
	Dictionary server	21 + *number of attributes* × 3 + (*number of user-defined types* × 6 + 2)[1] + 11[2] + (*number of routines that are invalid* × 4 + 2 + *number of resources of the routines that become invalid* × 5 + 1)[3] + *number of locked resources for function or procedure definitions*[4]
	Back-end server	*number of routines that become invalid*

[1] Add this if a user-defined type attribute is present.

[2] Add this if a subtype definition is involved.

[3] Add this if there is a routine that becomes invalid.

[4] Add this if a function or procedure is to be defined. Refer to the number of lock requests in the function definition or procedure definition.

(14) CREATE USER MAPPING

Condition	Number of lock requests
Dictionary server	11

(15) CREATE VIEW

Condition	Number of lock requests
HiRDB/Single Server	31 + *number of tables accessed in SELECT statement preprocessing* + *number of view tables accessed in SELECT statement preprocessing* + *number of base tables that are the basis of view tables accessed in SELECT statement preprocessing* + *number of columns* × 4 + *number of tables that are the basis of the view table* × 4 + *number of SELECT statements preprocessing*

Condition		Number of lock requests
HiRDB/Parallel Server	Front-end server	3 + *number of tables accessed in SELECT statement preprocessing* + *number of view tables accessed in SELECT statement preprocessing* + *number of base tables that are the basis of view tables accessed in SELECT statements preprocessing*
	Dictionary server	28 + *number of columns* × 4 + *number of tables that are the basis of the view table* × 4 + *number of SELECT statements preprocessing*

(16) DROP ALIAS

Condition	Number of lock requests
HiRDB/Single Server	10
HiRDB/Parallel Server (Dictionary server)	

(17) DROP CONNECTION SECURITY

Condition		Number of lock requests
HiRDB/Single Server		7 + *number-of-users-registered-in-dictionary-table-SQL_USERS*
HiRDB/Parallel Server	Front-end server	1
	Dictionary server	6 + *number-of-users-registered-in-dictionary-table-SQL_USERS*

(18) DROP DATA TYPE

Condition	Number of lock requests
HiRDB/Single Server	26 + *number of routines that become invalid* + *number of attributes* × 3 Add the following for a subtype: + 6 Add the following if there are user-defined type attributes: + *number of user-defined type attributes* × 4 Add the following if there are routines that become invalid: + *number of routines that become invalid* × 4 + 2 + *number of resources of routines that become invalid* × 5 + 1 Add the following if there are function definitions: + *number of function definitions* × 7 + 1 + *number of function parameters* × 4 + 1 + *number of procedures used by the functions* × 5 + 1 Add the following if there are procedure definitions: + *number of procedure definitions* × 5 + 1 + *number of procedure parameters* × 4 + 1 + *number of resources used by procedures* × 5 + 1 Add the following if a trigger action condition is specified: + 15 Add the following if there are trigger action procedures in the routines that become invalid: + *number of trigger action procedures* × 5

Condition		Number of lock requests
HiRDB/Parallel Server	Front-end server	1 + *number of routines that become invalid*
	Dictionary server	23 + *number of attributes* × 3 + *number of routines in which objects become invalid* Add the following for a subtype: + 6 Add the following if there are user-defined type attributes: + *number of user-defined type attributes* × 4 Add the following if there are routines that become invalid: + *number of routines that become invalid* × 4 + 2 + *number of resources of routines that become invalid* × 5 + 1 Add the following if there are function definitions: + *number of function definitions* × 7 + 1 + *number of function parameters* × 4 + 1 + *number of procedures used by the functions* × 5 + 1 Add the following if there are procedure definitions: + *number of procedure definitions* × 5 + 1 + *number of procedure parameters* × 4 + 1 + *number of resources used by procedures* × 5 + 1 Add the following if a trigger action condition is specified: + 15 Add the following if there are trigger action procedures in the routines that become invalid: + *number of trigger action procedures* × 5
	Back-end server	*number of routines in which objects become invalid*

(19) DROP FOREIGN INDEX

Condition	Number of lock requests
Front-end server	1
Dictionary server	21 + *number of member columns* × 3 Add the following when creating optimization information for a foreign index: + 4

(20) DROP FOREIGN TABLE

Condition	Number of lock requests
Front-end server	1 + *number of view tables defined based on foreign tables* + *number of routines that reference the foreign tables or view tables that are to be deleted*

Condition	Number of lock requests
Dictionary server	19 + *number of columns* × 4 + *number of privilege definitions for the foreign table to be deleted* × 3 + *number of routines in which objects become invalid* Add the following if indexes are defined: + *number of foreign indexes* × 5 + Σ (*number of member columns* × 7 + 4) Add the following when creating optimization information: + 2 + *number of columns for which optimization information is to be created* × 3 + *number of foreign indexes for which optimization information is to be created* × 3 Add the following for the base table of a view table: + *number of view tables to be deleted* + Σ {20 + *number of columns* × 4 + *number of base tables for view tables* + *number of privilege definitions for view tables* × 3} + 2 Add the following if there are routines that become invalid: + *number of routines in which objects become invalid* + Σ (2 + *number of resources inside a procedure* × 5) + 1
Back-end server	*number of routines in which objects become invalid*

(21) DROP FUNCTION

Condition		Number of lock requests
HiRDB/Single Server		*number of locked resources in DROP PROCEDURE* Add the following if there are parameters: + *number of parameters* × 2 Add the following if there are user-defined type parameters: + *number of user-defined types* × 2 Add the following if a trigger action condition is specified: + 15
HiRDB/Parallel Server	Front-end server and Back-end server	*same as DROP PROCEDURE*
	Dictionary server	*number of locked resources in DROP PROCEDURE* Add the following if there are parameters: + *number of parameters* × 2 Add the following if there are user-defined type parameters: + *number of user-defined types* × 2 Add the following if a trigger action condition is specified: + 15

(22) DROP INDEX (not a plug-in index)

Condition		Number of lock requests
HiRDB/Single Server*		22 + *number of routines in which objects or indexes become invalid* + *number of RDAREAs for index* × 7 + *number of member columns* × 3 + *number of index segments* Add the following if there are routines that become invalid: + *number of invalid routines* × 7 + 1 Add the following if there are trigger action procedures that reference the table for which indexes are defined: + *number of trigger action procedures that reference the table for which indexes are defined* × 5 Add the following if the index optimization information is collected: + 4 Add the following if the inner replica facility is used and a LOB column or LOB attribute is defined in the table for which indexes are defined: + *number of RDAREAs for LOB column* + *number of RDAREAs for LOB attribute* Add the following if the facility for predicting reorganization time is used: + *number of RDAREAs for index* × 62 + 1
HiRDB/Parallel Server	Front-end server	1 + *number of routines in which objects or indexes become invalid*
	Dictionary server	16 + *number of RDAREAs for index* × 3 + *number of member columns* × 3 + *number of routines in which objects or indexes become invalid* Add the following if there are routines that become invalid: + *number of invalid routines* × 7 + 1 Add the following if there are trigger action procedures that reference the table for which indexes are defined: + *number of trigger action procedures that reference the table for which indexes are defined* × 5 Add the following if the index optimization information is collected: + 4 Add the following if the facility for predicting reorganization time is used: + *number of RDAREAs for index* × 62 + 1
	Back-end server*	5 + *number of RDAREAs for index* × 4 + *number of index segments* + *number of routines in which objects or indexes become invalid* Add the following if the inner replica facility is used and a LOB column or LOB attribute is defined in the table for which indexes are defined: + *number of RDAREAs for LOB column* + *number of RDAREAs for LOB attribute*

Condition		Number of lock requests
	Reference-only back-end server*	2

* If the inner replica facility is used, a number of lock requests equaling the number of generations is required.

(23) DROP INDEX (plug-in index)

Condition		Number of lock requests
HiRDB/Single Server*		15 + *number of routines in which objects or indexes become invalid* + *number of RDAREAs storing the index* × 10 + *number of member columns* × 4 + *number of applicable functions* × 3 + *number of index segments* Add the following if the inner replica facility is used and a LOB column or LOB attribute is defined in the table for which indexes are defined: + *number of RDAREAs for LOB column* + *number of RDAREAs for LOB attribute* Add the following if the facility for predicting reorganization time is used: + *number of RDAREAs for index* × 62 + 1
HiRDB/Parallel Server	Front-end server	1 + *number of routines in which objects or indexes become invalid*
	Dictionary server	9 + *number of RDAREAs storing the index* × 6 + *number of member columns* × 4 + *number of applicable functions* × 3 + *number of routines in which objects or indexes become invalid* Add the following if the facility for predicting reorganization time is used: + *number of RDAREAs for index* × 62 + 1
	Back-end server*	5 + *number of RDAREAs storing the index* × 4 + *number of index segments* + *number of routines in which objects or indexes become invalid* Add the following if the inner replica facility is used and a LOB column or LOB attribute is defined in the table for which indexes are defined: + *number of RDAREAs for LOB column* + *number of RDAREAs for LOB attribute*

* If the inner replica facility is used, a number of lock requests equaling the number of generations is required.

(24) DROP PROCEDURE

Condition		Number of lock requests
HiRDB/Single Server		19 + *number of parameters* × 5 + *number of resources* × 10 + *number of routines in which objects become invalid* Add the following if there are trigger action procedures in the routines that become invalid: + *number of trigger action procedures* × 5
HiRDB/Parallel Server	Front-end server	1 + *number of routines in which objects become invalid*
	Dictionary server	18 + *number of parameters* × 3 + *number of resources* × 10 + *number of routines in which objects become invalid* Add the following if there are trigger action procedures in the routines that become invalid: + *number of trigger action procedures* × 5
	Back-end server	*number of routines in which objects become invalid*

(25) DROP SCHEMA

Condition	Number of lock requests
HiRDB/Single Server[1]	11 + *number of tables inside the schema* + *number of view tables inside the schema* + *number of routines inside the schema* + *number of view tables in other schemas that use the tables or view tables inside the schema as base tables* + *number of indexes in other schemas that reference the tables, view tables, routines, or data types to be deleted* Add the number of tables in the schema that are to be deleted (when they are not matrix-partitioned tables). + Σ {*number of RDAREAs for table* × 3 + *number of specified partitioning conditions* × 9 + *number of columns* × 4 + (*number of LOB columns* × 4) × *number of RDAREAs for table* + 1 + 8} + 2

Condition	Number of lock requests
	Add the number of tables in the schema that are to be deleted (when they are matrix-partitioned tables). + Σ {*number of RDAREAs for table* × 3 + *number of specified RDAREAs for table* (including overlapped ones) × 12 + *number of columns* × 4 + (*number of LOB columns* × 4) × *number of RDAREAs for table* + 1 + 8 + *number of privilege definitions for the table to be deleted* + *number of partitioning keys* × 2 + (*number of specified storage conditions* + 2) × 2 + 2} + 3 Add the number of indexes in the schema that are to be deleted. + 3 + Σ (*number of RDAREAs for index* × 7 + *number of member columns* × 3) + Σ (*number of RDAREAs for table* × 10 + 10 + *number of table data segments being used*) + Σ (*number of RDAREAs for index* × 8 + 10 + *number of index segments being used*) + 8 Add the following if the index optimization information is collected: + *number of tables for which optimization information is collected* × 2 + *number of columns for which optimization information is collected* × 3 + *number of indexes for which optimization information is collected* × 3 Add the number of view tables that are deleted along with the tables being deleted: + 2 + Σ (12 + *number of base tables for the view tables* × 4 + *number of columns* × 4) Add the number in the routines that use the tables, view tables, routines, or data types inside the schema: + 3 + Σ (5 + *number of resources inside the procedure* × 5 + *number of defined parameters* × 3) Add the following if there are procedures that become invalid: + *number of invalid procedures* + Σ (2 + *number of resources* × 5) + number of requests for the data types defined in the schema[2] Add the following if there are trigger action procedures that reference the tables to be deleted: + *number of trigger action procedures that reference the tables to be deleted* × 5

Condition		Number of lock requests
		Add the following if there are tables for which triggers are defined: + *number of triggers defined in the schema* × 8 + *total number of columns specified in the UPDATE trigger definition in the schema* × 3 + *number of triggers that have trigger action conditions in the schema* × 3 + *number of resource types referenced in the trigger action conditions in the schema* × 5 Add the following if a referential constraint is defined: + *number of referenced tables* + *number of foreign keys* × 10 Add the following if a check constraint is defined: + *number of check constraints* × 9 Add the following if the facility for predicting reorganization time is used: + Σ {(*number of RDAREAs for table* + (*number of LOB columns* + *number of LOB attributes*) × *number of RDAREAs for table*) × 62} + Σ {(*number of RDAREAs for index* + (*number of LOB columns* + *number of LOB attributes*) × *number of RDAREAs for table*) × 62} + 1
HiRDB/Parallel Server	Front-end server	3 + *number of tables in the schema* + *number of foreign tables in the schema* + *number of view tables in the schema* + *number of routines in the schema* + *number of view tables in other schemas that use the tables, foreign tables or view tables inside the schema as base tables* + *number of routines in other schemas that reference the tables, foreign tables, view tables, routines, or data types to be deleted*
	Dictionary server	11 + *number of indexes in other schemas* Add the number of tables in the schema that are to be deleted (when they are not matrix-partitioned tables). + Σ {*number of RDAREAs for table* × 3 + *number of specified partitioning conditions* × 9 + *number of columns* × 4 + (*number of LOB columns* × 4) × *number of RDAREAs for table* + *number of privilege definitions for the table to be deleted* × 3 + 6} + 11 Add the number of tables in the schema that are to be deleted (when they are matrix-partitioned tables). + Σ {*number of RDAREAs for table* × 3 + *number of specified RDAREAs for table* (including the overlapped ones) × 12 + *number of columns* × 4 + (*number of LOB columns* × 4) × *number of RDAREAs for table* + *number of privilege definitions for the table to be deleted* × 3 + 1 + 6 + *number of partitioning keys* × 2 + (*number of specified storage conditions* + 2) × 2 + 2} + 3

Condition		Number of lock requests
		Add the number of tables in the schema that are to be deleted (foreign tables). + Σ (*number of columns* × 4 + *number of privilege definitions for the foreign table to be deleted* × 3 + 4) + 8 Add the number of indexes and foreign indexes in the schema to be deleted. + 4 + Σ (*number of RDAREAs for indexes* × 7 + *number of member columns* × 7) + *number of indexes* × 5 + *number of foreign indexes* × 5 Add the following if the optimization information is collected: + *number of tables for which optimization information is collected* × 2 + *number of columns for which optimization information is collected* × 3 + *number of foreign tables for which optimization information is collected* × 2 + *number of foreign table columns for which optimization information is collected* × 3 + *number of indexes for which optimization is collected* × 3 + *number of foreign indexes for which optimization information is collected* × 3 Add the requests for the view tables that are deleted when tables or foreign tables are deleted. + 2 + Σ (18 + *number of base tables for the view tables* × 4 + *number of columns* × 4 + *number of privilege definitions for the view tables* × 3) Add the requests for the routines that use the tables, foreign tables, view tables, routines, and data types in the schema. + 3 + Σ (5 + *number of resources inside the procedure* × 10 + *number of parameter definitions* × 3) Add the following if there are procedures that become invalid: + *number of invalid procedures* + Σ (5 + *number of resources* × 5) + *number of requests for the data types defined in the schema*[2] Add the following if there are trigger action procedures that reference the tables to be deleted: + *number of trigger action procedures that reference the tables to be deleted* × 5

Condition		Number of lock requests
		Add the following if there are tables for which triggers are defined: + *number of triggers defined in the schema* × 8 + *total number of columns specified in the UPDATE trigger definition in the schema* × 3 + *number of triggers that have trigger action conditions in the schema* × 3 + *number of resource types referenced in the trigger action conditions in the schema* × 5 Add the following if a referential constraint is defined: + *number of referenced tables* + *number of foreign keys* × 10 Add the following if a check constraint is defined: + *number of check constraints* × 9 Add the following if the facility for predicting reorganization time is used: + Σ {(*number of RDAREAs for table* + (*number of LOB columns* + *number of LOB attributes*) × *number of RDAREAs for table*) × 62} + Σ {(*number of RDAREAs for index* + (*number of LOB columns* + *number of LOB attributes*) × *number of RDAREAs for table*) × 62} + 1
	Back-end server[1]	+ Σ (*number of RDAREAs for table* × 10 + 10 + *number of table data segments being used*) + Σ (*number of RDAREAs for index* × 8 + 10 + *number of index segments being used*) + 8 + *number of routines in which objects or index objects become invalid*
	Reference-only back-end server[1]	*Number of tables* + *number of RDAREAs for index*

[1] If the inner replica facility is used, a number of lock requests equaling the number of generations is required.

[2] See the number of lock requests for user-defined deletion.

(26) DROP SERVER

Condition	Number of lock requests
Dictionary server	9 + *number of privilege definitions for the foreign server to be deleted*

(27) DROP TABLE

Condition	Number of lock requests
HiRDB/Single Server[1]	Add the following if the table is not a matrix-partitioned table: 24 + *number of view tables defined based on the table* + *number of routines that reference the table or view table to be deleted* + *number of RDAREAs for table* × 3 + *number of specified partitioning conditions* × 9 + *number of columns* × 4 + 5 + *number of RDAREAs for table* × 4 + *number of table data segments being used* Add the following if the table is a matrix-partitioned table: 22 + *number of view tables defined based on the table* + *number of routines that reference the table or view table to be deleted* + *number of RDAREAs for table* × 3 + *number of RDAREAs for table* (including overlapped ones) × 8 + *number of columns* × 4 + 5 + *number of RDAREAs for table* × 4 + *number of table data segments being used* + *number of partitioning keys* × 2 + (*number of specified storage conditions* + 2) × 2 + 2 Add the following if indexes are defined: + Σ (*number of RDAREAs for index* × 2) + *number of index segments being used* + Σ {(*number of RDAREAs for index* × 7) + (*number of member columns* × 3)} + 3 Add the following if LOB columns are defined: + (*number of LOB columns* × 4) × *number of RDAREAs for table* + 1 Add the following if the index optimization information is collected: + 2 + *number of columns for which optimization information is collected* × 3 + *number of indexes for which optimization information is collected* × 3 Add the following for the base tables of view tables: + *number of view tables to be deleted* + Σ {(12 + *number of base tables for view tables*) + *number of columns* × 4} + 2 Add the following if there are routines that become invalid: + *number of routines in which objects become invalid* + Σ (2 + *number of resources inside the procedure* × 5) Add the following if there are trigger action procedures that reference the tables to be deleted: + *number of trigger action procedures that reference the tables to be deleted* × 5 Add the following if user-defined columns are defined: + *number of user-defined columns* × 4 + *number of LOB attributes* × *number of partitioned RDAREAs* × 5 + *number of abstract data-type attributes* + 4

Condition		Number of lock requests
		Add the following if triggers are defined: + *number of defined triggers* × 8 + *total number of columns specified in the UPDATE trigger definition* × 3 + *number of triggers that have trigger action conditions* × 3 + *number of resource types referenced in the trigger action conditions* × 5 Add the following if a user-defined type that includes a LOB column or LOB attribute is defined: + Σ (↑ *number of segments allocated to RDAREAs* ÷ 64000 ↑ + 1)^2 Add the following if a referential constraint is defined: + *number of referenced tables* + *number of foreign keys* × 10 + *total number of functions in referenced tables that are affected by deletion of constraint* Add the following if CASCADE is defined for referential constraint operation: +*number of CASCADEs* × 17 +*number of referenced tables specified by CASCADE* × 4 Add the following if the specification of referential constraint operation is UPDATE ON CASCADE: + *total number of primary key member columns of the referenced table for which CASCADE is specified* × 8 + *number of referenced tables specified by CASCADE* × 3 Add the following if a check constraint is defined: + *number of check constraints* × 9 Add the following if the facility for predicting reorganization time is used: + {*number of RDAREAs for table* + *number of RDAREAs for index* + (*number of LOB columns* + *number of LOB attributes*) × *number of RDAREAs for table*} × 62 + 1
HiRDB/Parallel Server	Front-end server	3 + *number of view tables defined based on tables* + *number of routines that reference the table or view table to be deleted* Add the following if a referential constraint is defined: + *number of referenced tables* × 4 + *total number of functions in referenced tables that are affected by deletion of constraint*

Condition		Number of lock requests
	Dictionary server	Add the following if the table is not a matrix-partitioned table: 21 + *number of RDAREAs for table* × 3 + *number of specified partitioning conditions* × 9 + *number of columns* × 4 + *number of routines in which objects become invalid* Add the following if the table is a matrix-partitioned table: 19 + *number of RDAREAs for table* × 3 + *number of specified partitioning conditions* × 9 + *number of columns* × 4 + *number of routines in which objects become invalid* Add the following if indexes are defined: + Σ {(*number of RDAREAs for index* × 7) + *number of member columns* × 3} + 3 Add the following if LOB columns are defined: + (*number of LOB columns* × 4) × *number of RDAREAs for table* + 1 Add the following if indexes are defined: + Σ {(*number of RDAREAs for index* × 7) + *number of member columns* × 3} + 3 Add the following if the index optimization information is collected: + 2 + *number of columns for which optimization information is collected* × 3 + *number of indexes for which optimization information is collected* × 3 Add the following for the base tables of view tables: + *number of view tables to be deleted* + Σ {12 + *number of base tables for view tables*) + *number of columns* × 4} + 2 Add the following if there are routines that become invalid: + *number of routines that become invalid* + Σ (2 + *number of resources inside the procedure* × 5) Add the following if there are trigger action procedures that reference the tables to be deleted: + *number of trigger action procedures that reference the tables to be deleted* × 5 Add the following if user-defined columns are defined: + *number of user-defined columns* × 4 + *number of LOB attributes* × *number of partitioned RDAREA* × 5 + *number of abstract data-type attributes* + 4 Add the following if triggers are defined: + *number of defined triggers* × 8 + *total number of columns specified in the UPDATE trigger definition* × 3 + *number of triggers that have trigger action conditions* × 3 + *number of resource types referenced in the trigger action conditions* × 5

Condition		Number of lock requests
		Add the following if a referential constraint is defined: + *number of referenced tables* + *number of foreign keys* × 10 Add the following if CASCADE is defined for referential constraint operation: +*number of CASCADEs* × 17 +*number of referenced tables specified by CASCADE* × 4 Add the following if the specification of referential constraint operation is UPDATE ON CASCADE: + *total number of primary key member columns of the referenced table for which CASCADE is specified* × 8 + *number of referenced tables specified by CASCADE* × 3 Add the following if a check constraint is defined: + *number of check constraints* × 9 Add the following if the facility for predicting reorganization time is used: + {*number of RDAREAs for table* + *number of RDAREAs for index* + (*number of LOB columns* + *number of LOB attributes*) × *number of RDAREAs for table*} × 62 + 1
	Back-end server[1]	5 + *number of RDAREAs for table* × 4 + *number of table data segments being used* + *number of routines in which objects become invalid* Add the following if indexes are defined: + Σ (*number of RDAREAs for index* × 2) + *number of index segments being used* Add the following if a user-defined type that includes a LOB column or LOB attribute is defined: + Σ (↑ *number of segments allocated to RDAREAs* ÷ 64000 ↑ + 1)[2]
	Reference-only back-end server[1]	2 Add the following if shared indexes are defined: + *number of RDAREAs for index*

[1] If the inner replica facility is used, a number of lock requests equaling the number of generations is required.

[2] Repeat and add this for the number of LOB RDAREAs.

(28) DROP TRIGGER

Condition		Number of lock requests
HiRDB/Single Server		19 + *number of routines in which objects become invalid* + *number of resources used by trigger action procedures* × 5 Add the following if a trigger action condition is specified: + *number of resources used inside trigger action condition specification* × 5 + 3 Add the following if a column name modified by an old or new values correlation name is used inside a trigger action procedure: + *number of column types modified by new values correlation names* × 3 + *number of column types modified by old values correlation names* × 3 + 3 Add the following if trigger event columns are specified: + *number of trigger event columns* × 3 Add the following if there are routines that become invalid: + *number of invalid routines* × 5 + *number of resources used by routines* × 5 Add the following if there are trigger action procedures in the routines that become invalid: + *number of trigger action procedures* × 5
HiRDB/Parallel Server	Front-end server	3 + *number of routines in which objects become invalid*
	Dictionary server	16 + *number of routines in which objects become invalid* + *number of resources used by trigger action procedures* × 5 Add the following if a trigger action condition is specified: + *number of resources used inside trigger action condition specification* × 5 + 3 Add the following if a column name modified by an old or new values correlation name is used inside a trigger action procedure: + *number of column types modified by new values correlation names* × 3 + *number of column types modified by old values correlation names* × 3 + 3 Add the following if trigger event columns are specified: + *number of trigger event columns* × 3 Add the following if there are routines that become invalid: + *number of invalid routines* × 5 + *number of resources used by routines* × 5 Add the following if there are trigger action procedures in the routines that become invalid: + *number of trigger action procedures* × 5
	Back-end server	*number of routines in which objects become invalid*

(29) DROP USER MAPPING

Condition	Number of lock requests
Dictionary server	9

(30) DROP VIEW

Condition		Number of lock requests
HiRDB/Single Server		10 + *number of view tables to be deleted* + *number of procedures that become invalid* + Σ (15 + *number of columns* × 4 + *number of tables that are the basis of the view table* × 4) Add the following if there are procedures that become invalid: + *number of procedures that become invalid* + Σ (2 + *number of resources in procedures* × 5) Add the following if there are trigger action procedures that reference the view tables to be deleted: + *number of trigger action procedures that reference the view tables to be deleted* × 5
HiRDB/Parallel Server	Front-end server	*number of view tables to be deleted* + *number of procedures that become invalid*
	Dictionary server	10 + Σ (15 + *number of columns* × 4 + *number of tables that are the basis of the view table* × 4) Add the following if there are routines that become invalid: + *number of routines that become invalid* + Σ (2 + *number of resources in routines* × 5) Add the following if there are trigger action procedures that reference the view tables to be deleted: + *number of trigger action procedures that reference the view tables to be deleted* × 5
	Back-end server	*number of routines in which objects become invalid*

E.2 Data manipulation SQLs

(1) SELECT (LOCK TABLE not specified, WITHOUT LOCK not specified)

Condition		**Number of lock requests**
HiRDB/Single Server		[preprocessing][1] + *number of RDAREAs for table to be retrieved* + *number of tables* + *number of rows with retrieval hit*[3] + *number of RDAREAs storing the indexes used*[2] + *number of index keys used in retrieval*[5] Add the following if an index-type plug-in is used for retrieval or if a data-type plug-in is defined in the columns to be referenced: + *number of logical files used by the plug-in*[4] The following should be noted if table data is retrieved via a list: • The number of lock requests in preprocessing in the formula should be determined by applying it to the base table of the list. • The n*umber of RDAREAs storing the tables to be retrieved* and the *number of tables in the formula should be determined for* the base table of the list. • The following value must be added: + 1 Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for retrieval: + *number of routines used*
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing][1] If table data is retrieved via a list, the number of lock requests in preprocessing in the formula should be determined by applying it to the base table of the list. Add the following if routines are used for retrieval: + *number of routines used*
	Back-end server	*Number of RDAREAs for table to be retrieved* + *number of tables* + *number of rows with retrieval hit*[3] + *number of RDAREAs storing the indexes used*[2] + number of indexes used in retrieval[5] Add the following if an index-type plug-in is used for retrieval or if a data-type plug-in is defined in the columns to be referenced: + *number of logical files used by the plug-in*[4] The following should be noted if table data is retrieved via a list: • The n*umber of RDAREAs storing the tables to be retrieved* and the *number of tables in the formula should be determined for* the base table of the list. • The following value must be added: + 1 Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for retrieval: + *number of routines used*

[1] For the number of lock requests in preprocessing, see (25).

[2] If indexes are defined for columns specified with AND or OR in a retrieval condition expression, those indexes will be used as a rule.

[3]

- If data is to be retrieved from multiple tables, determine the number of hit selection rows in individual tables and add that number.
- If two or more conditions are specified for index definition columns (one or more if `FOR UPDATE` is specified), determine the combined total of the rows that are retrieved based only on the conditions for individual indexes.
- Determine the number of hit selection rows after excluding conditions that use the columns of multiple tables and conditions that use functions.
- Use the number of hit selection pages if a lock is specified for each page.

[4] For the number of logical files used by plug-ins, see the manuals included with the plug-ins.

[5] The value is 1 if index key value lock is specified, and 0 if index key value lock is not specified.

(2) SELECT (LOCK TABLE not specified, WITHOUT LOCK WAIT specified)

Condition	Number of lock requests
HiRDB/Single Server	[preprocessing][1] + *number of RDAREAs to be retrieved* + *number of tables* + 1 + *number of RDAREAs for index used*[2] + *number of index keys used in retrieval*[4] Add the following if an index-type plug-in is used for retrieval or if a data-type plug-in is defined in the columns to be referenced: + *number of logical files used by the plug-in*[3] The following should be noted if table data is retrieved via a list: • The number of lock requests in preprocessing in the formula should be determined by applying it to the base table of the list. • The n*umber of RDAREAs storing the tables to be retrieved* and the *number of tables in the formula should be determined for* the base table of the list. • The following value must be added: + 1 Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for retrieval: + *number of routines used*

Condition		Number of lock requests
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing][1] If table data is retrieved via a list, the number of lock requests in preprocessing in the formula should be determined by applying it to the base table of the list. Add the following if routines are used for retrieval: + *number of routines used*
	Back-end server	*Number of RDAREAs to be retrieved* + *number of tables* + 2 + *number of RDAREAs for index used*[2] + *number of index keys used in retrieval*[4] Add the following if an index-type plug-in is used for retrieval or if a data-type plug-in is defined in the columns to be referenced: + *number of logical files used by the plug-in*[3] The following should be noted if table data is retrieved via a list: • The n*umber of RDAREAs storing the tables to be retrieved* and the *number of tables in the formula should be determined for* the base table of the list. • The following value must be added: + 1 Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for retrieval: + *number of routines used*

[1] For the number of lock requests in preprocessing, see (25).

[2] If indexes are defined for columns specified with AND or OR in a retrieval condition expression, those indexes will be used as a rule.

[3] For the number of logical files used by plug-ins, refer to the manuals included with the plug-ins.

[4] The value is 1 if index key value lock is specified, and 0 if index key value lock is not specified.

(3) SELECT (LOCK TABLE not specified, WITHOUT LOCK NOWAIT specified)

Condition	Number of lock requests
HiRDB/Single Server	[preprocessing]^1 + *number of RDAREAs to be retrieved* + *number of tables* + *number of RDAREAs storing indexes used*^{2} Add the following if an index-type plug-in is used for retrieval or if a data-type plug-in is defined in the columns to be referenced: + *number of logical files used by the plug-in*^{3} The following should be noted if table data is retrieved via a list: • The number of lock requests in preprocessing in the formula should be determined by applying it to the base table of the list. • The n*umber of RDAREAs storing the tables to be retrieved* and the *number of tables in the formula should be determined for* the base table of the list. • The following value must be added: + 1 Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if LOCK is specified for the pd_nowait_scan_option operand (add only for a non-FIX table): + 1 Add the following if routines are used for retrieval: + *number of routines used*

Condition		Number of lock requests
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing][1] If table data is retrieved via a list, the number of lock requests in preprocessing in the formula should be determined by applying it to the base table of the list. Add the following if routines are used for retrieval: + *number of routines used*
	Back-end server	*Number of RDAREAs to be retrieved* + *number of tables* + *number of RDAREAs storing indexes used*[2] Add the following if an index-type plug-in is used for retrieval or if a data-type plug-in is defined in the columns to be referenced: + *number of logical files used by the plug-in*[3] The following should be noted if table data is retrieved via a list: • The n*umber of RDAREAs storing the tables to be retrieved* and the *number of tables in the formula should be determined for* the base table of the list. • The following value must be added: + 1 Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if LOCK is specified for the pd_nowait_scan_option operand (add only for a non-FIX table): + 1 Add the following if routines are used for retrieval: + *number of routines used*

[1] For the number of lock requests in preprocessing, see (25).

[2] If indexes are defined for columns specified with AND or OR in a retrieval condition expression, those indexes will be used as a rule.

[3] For the number of logical files used by plug-ins, refer to the manuals included with the plug-ins.

(4) SELECT (LOCK TABLE specified[1])

Condition		Number of lock requests
HiRDB/Single Server		[preprocessing][2] + *number of RDAREAs to be retrieved* + *number of RDAREAs storing indexes used* Add the following if an index-type plug-in is used for retrieval or if a data-type plug-in is defined in the columns to be referenced: + *number of logical files used by the plug-in*[3] The following should be noted if table data is retrieved via a list: • The number of lock requests in preprocessing in the formula should be determined by applying it to the base table of the list. • The n*umber of RDAREAs storing the tables to be retrieved* and the *number of tables in the formula should be determined for* the base table of the list. • The following value must be added: + 1 Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for retrieval: + *number of routines used*
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing][2] If table data is retrieved via a list, the number of lock requests in preprocessing in the formula should be determined by applying it to the base table of the list. Add the following if routines are used for retrieval: + *number of routines used*
	Back-end server	*Number of RDAREAs to be retrieved* + *number of RDAREAs storing indexes used* Add the following if an index-type plug-in is used for retrieval or if a data-type plug-in is defined in the columns to be referenced: + *number of logical files used by the plug-in*[3] The following should be noted if table data is retrieved via a list: • The n*umber of RDAREAs storing the tables to be retrieved* and the *number of tables in the formula should be determined for* the base table of the list. • The following value must be added: + 1 Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for retrieval: + *number of routines used*

[1] If `LOCK TABLE IN SHARE MODE` is specified for `SELECT` with `FOR UPDATE` specified, the same formula should be used as when `LOCK TABLE` in not specified.

[2] For the number of lock requests in preprocessing, see (25).

[3] For the number of logical files used by plug-ins, refer to the manuals included with the plug-ins.

(5) INSERT (INSERT to VALUES clauses specified, and LOCK TABLE not specified)

Condition		Number of lock requests
HiRDB/Single Server		[preprocessing][1] + 5 + *number of RDAREAs to be inserted*[3] + *number of inserted VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of branched rows* + *number of inserted BINARY-type columns that cannot be stored in one page* × *number of branched rows* + *number of indexes* Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by plug-ins*[2] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for insertion: + *number of routines used*
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing][1] Add the following if routines are used for insertion: + *number of routines used*
	Back-end server	5 + *number of RDAREAs to be inserted*[3] + *number of inserted VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of branched rows* + *number of inserted BINARY-type columns that cannot be stored in one page* × *number of branched rows* + *number of indexes* Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by plug-ins*[2] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for insertion: + *number of routines used*

[1] For the number of lock requests in preprocessing, see (25).

[2] For the number of logical files used by plug-ins, refer to the manuals included with the plug-ins.

[3] Total number of RDAREAs for the target tables, indexes, LOB columns, and LOB attributes.

(6) INSERT (INSERT to VALUES clauses specified, LOCK TABLE specified[1])

Condition		Number of lock requests
HiRDB/Single Server		[preprocessing][2] + 1 + *number of RDAREAs to be inserted*[4] Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by plug-in*[*3] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for insertion: + *number of routines used*
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing][2] Add the following if routines are used for insertion: + *number of routines used*
	Back-end server	1 + *number of RDAREAs to be inserted*[*4] Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by plug-ins*[*3] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for insertion: + *number of routines used*

[1] If `LOCK TABLE IN SHARE MODE` is specified, the same formula should be used as when `LOCK TABLE` in not specified.

[2] For the number of lock requests in preprocessing, see (25).

[3] For the number of logical files used by plug-ins, refer to the manuals included with the plug-ins.

[4] Total number of RDAREAs for the target indexes, LOB columns, and LOB attributes.

(7) INSERT (INSERT to SELECT clauses specified)

Condition		Number of lock requests
HiRDB/Single Server		[preprocessing][1] + 5 + *number of RDAREAs to be inserted*[5] + 4 + *number of inserted rows*[2] + *number of inserted VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of inserted rows*[2] × *number of branched rows* + *number of inserted BINARY-type columns that cannot be stored in one page* × *number of inserted rows*[2] × *number of branched rows* + *number of inserted rows* × *number of indexes* + 1 + *number of RDAREAs to be retrieved*[5] + *number of tables to be retrieved* + *number of hit selection rows*[3] Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by the plug-ins to be inserted*[4] + *number of logical files used by the plug-ins used for retrieval*[4] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for insertion: + *number of routines used*
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing][1] Add the following if routines are used for insertion: + *number of routines used*
	Back-end server	4 + *number of RDAREAs to be inserted*[5] + *number of inserted rows*[2] + *number of inserted VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of inserted rows* × *number of inserted BINARY-type columns that cannot be stored in one page* × *number of inserted rows* × *number of branched rows* + *number of inserted rows* × *number of indexes* + 1 + *number of RDAREAs to be retrieved*[5] + *number of tables to be retrieved* + *number of hit selection rows*[3] Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by the plug-ins to be inserted*[4] + *number of logical files used by the plug-ins used for retrieval*[4] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for insertion: + *number of routines used*

[1] For the number of lock requests in preprocessing, see (25).

[2] Use the number of pages inserted if locking by page is specified.

[3]

- If data is to be retrieved from multiple tables, determine the number of hit selection rows in individual tables and add that number.

- If two or more conditions are specified for index definition columns, determine the combined total of the rows that are retrieved based only on the conditions for individual indexes.
- Determine the number of hit selection rows after excluding conditions that use the columns of multiple tables and conditions that use functions.
- Use the number of hit selection pages if a lock is specified for each page.

[4] For the number of logical files used by plug-ins, refer to the manuals included with the plug-ins.

[5] Total number of RDAREAs for the target tables, indexes, LOB columns, and LOB attributes.

(8) INSERT (INSERT to VALUES clauses specified, LOCK TABLE not specified, index key value no locking used)

Condition	Number of lock requests
HiRDB/Single Server	[preprocessing][1] + 5 + *number of RDAREAs to be inserted*[2] + *number of inserted VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of branched rows* + *number of inserted BINARY-type columns that cannot be stored in one page* × *number of branched rows* + *number of indexes* + 1 Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by plug-ins*[3] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for insertion: + *number of routines used*
HiRDB/Parallel Server	

[1] For the number of lock requests in preprocessing, see (25).

[2] Total number of RDAREAs for the target tables, indexes, LOB columns, and LOB attributes.

[3] For the number of logical files used by each plug-in, refer to the manual provided with the plug-in.

(9) INSERT (INSERT to VALUES clauses specified, LOCK TABLE specified, index key value no locking used)

Condition	Number of lock requests
HiRDB/Single Server	[preprocessing]^1 + 1 + *number of RDAREAs to be inserted*^{2} Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by plug-ins*^{3} Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for insertion: + *number of routines used*
HiRDB/Parallel Server	

[1] For the number of lock requests in preprocessing, see (25).

[2] Total number of RDAREAs for the target indexes, LOB columns, and LOB attributes.

[3] For the number of logical files used by each plug-in, refer to the manual provided with the plug-in.

(10) INSERT (INSERT to SELECT clauses specified, index key value no locking used)

Condition	Number of lock requests
HiRDB/Single Server	[preprocessing]^1 + *number of RDAREAs to be inserted*^{2} + 4 + *number of inserted rows*^{3} + *number of inserted VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of inserted rows* × *number of branched rows* + *number of inserted BINARY-type columns that cannot be stored in one page* × *number of inserted rows* × *number of branched rows* + *number of indexes* + 1 + *number of RDAREAs to be retrieved*^{2} + *number of tables to be retrieved* + *number of hit selection rows*^{4} Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by the plug-ins to be inserted*^{5} + *number of logical files used by the plug-ins used for retrieval*^{5} Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for insertion: + *number of routines used*
HiRDB/Parallel Server	

[1] For the number of lock requests in preprocessing, see (25).

[2] Total number of RDAREAs for the target tables, indexes, LOB columns, and LOB attributes.

[3] Use the number of inserted pages if a lock is specified for each page.

[4]

- If data is to be retrieved from multiple tables, determine the number of hit selection rows in individual tables and add that number.
- If two or more conditions are specified for index definition columns, determine the combined total of the rows that are retrieved based only on the conditions for individual indexes.
- Determine the number of hit selection rows after excluding conditions that use the columns of multiple tables and conditions that use functions.
- Use the number of hit selection pages if a lock is specified for each page.

[5] For the number of logical files used by each plug-in, refer to the manual provided with the plug-in.

(11) UPDATE (LOCK TABLE not specified)

Condition	Number of lock requests
HiRDB/Single Server	[preprocessing][1] + *number of RDAREAs to be updated*[5] + 4 + *number of updated rows*[2] + *number of pre-update VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of pre-update rows* × *number of branched rows before update* + *number of post-update VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of post-update rows* × *number of branched rows after update* + *number of pre-update BINARY-type columns that cannot be stored in one page* × *number of pre-update rows* × *number of branched rows before update* + *number of post-update BINARY-type columns that cannot be stored in one page* × *number of post-update rows* × *number of branched rows after update* + *number of indexes to be updated* × 2 × *number of updated rows* + Σ (*number of segments storing pre-update LOB data*)[4] Add the following if a data-type plug-in or index-type plug-in is defined for the columns to be updated: + *number of logical files used by the plug-ins*[3] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if LOCK is specified for the pd_nowait_scan_option operand (add only for a non-FIX table): + 1 Add the following if routines are used for updating: + *number of routines used*

Condition		Number of lock requests
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing][1] Add the following if routines are used for updating: + *number of routines used*
	Back-end server	*number of RDAREAs to be updated*[5] + 4 + *number of updated rows*[2] + *number of pre-update VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of pre-update rows* × *number of branched rows before update* + *number of post-update VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of post-update rows* × *number of branched rows after update* + *number of pre-update BINARY-type columns that cannot be stored in one page* × *number of pre-update rows* × *number of branched rows before update* + *number of post-update BINARY-type columns that cannot be stored in one page* × *number of post-update rows* × *number of branched rows after update* + *number of indexes to be updated* × 2 × *number of updated rows* + Σ (*number of segments storing pre-update LOB data*)[4] Add the following if a data-type plug-in or index-type plug-in is defined for the columns to be updated: + *number of logical files used by the plug-ins*[3] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if LOCK is specified for the pd_nowait_scan_option operand (add only for a non-FIX table): + 1 Add the following if routines are used for updating: + *number of routines used*

[1] For the number of lock requests in preprocessing, see (25).

[2]

- If there is a conditional expression that needs to retrieve data from another table, determine the number of hit selection rows for that table and add it to the number of updated rows.
- If conditions are specified for an index definition column, determine the combined total of the rows that are retrieved based only on the conditions for individual indexes. Furthermore, use the number of hit selection rows as the number of updated rows.
- Determine the number of hit selection rows after excluding conditions that use the columns of multiple tables and conditions that use functions. Furthermore, use the number of hit selection rows as the number of updated rows.
- Use the number of updated pages if a lock is specified for each page.

[3] For the number of logical files used by plug-ins, refer to the manuals included with the plug-ins.

[4] Repeat and add this for the number of LOB columns and LOB attributes.

[5] Total number of RDAREAs for the target tables, indexes, LOB columns, and LOB attributes.

(12) UPDATE (LOCK TABLE specified[1])

Condition		Number of lock requests
HiRDB/Single Server		[preprocessing][2] + *number of RDAREAs to be updated*[5] + 1 + Σ (*number of segments storing pre-update LOB data*)[4] Add the following if a data-type plug-in or index-type plug-in is defined for the columns to be updated: + *number of logical files used by the plug-ins*[3] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if LOCK is specified for the pd_nowait_scan_option operand (add only for a non-FIX table): + 1 Add the following if routines are used for updating: + *number of routines used*
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing][2] Add the following if routines are used for updating: + *number of routines used*
	Back-end server	*number of RDAREAs to be updated*[5] + 1 + Σ (*number of segments storing pre-update LOB data*)[4] Add the following if a data-type plug-in or index-type plug-in is defined for the columns to be updated: + *number of logical files used by the plug-ins*[3] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if LOCK is specified for the pd_nowait_scan_option operand (add only for a non-FIX table): + 1 Add the following if routines are used for updating: + *number of routines used*

[1] If `LOCK TABLE IN SHARE MODE` is used, use the same formula as that used when there is no `LOCK TABLE`.

[2] For the number of lock requests in preprocessing, see (25).

[3] For the number of logical files used by plug-ins, refer to the manuals included with the plug-ins.

[4] Repeat and add this for the number of LOB columns and LOB attributes.

[5] Total number of RDAREAs for the target indexes, LOB columns, and LOB attributes.

(13) UPDATE (LOCK TABLE not specified, index key value no locking used)

Condition	Number of lock requests
HiRDB/Single Server	[preprocessing][1] + *number of RDAREAs to be updated*[2] + 4 + *number of updated rows*[3] + *number of pre-update VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of pre-update rows* × *number of branched rows before update* + *number of post-update VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of post-update rows* × *number of branched rows after update* + *number of pre-update BINARY-type columns that cannot be stored in one page* × *number of pre-update rows* × *number of branched rows before update* + *number of post-update BINARY-type columns that cannot be stored in one page* × *number of post-update rows* × *number of branched rows after update* + Σ (*number of segments storing pre-update LOB data*)[4] Add the following if a data-type plug-in or index-type plug-in is defined for the columns to be updated: + *number of logical files used by plug-ins*[5] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if LOCK is specified for the pd_nowait_scan_option operand (add only for a non-FIX table): + 1 Add the following if routines are used for updating: + *number of routines used*
HiRDB/Parallel Server	

[1] For the number of lock requests in preprocessing, see (25).

[2] Total number of RDAREAs for the target tables, indexes, LOB columns, and LOB attributes.

[3]

- If there is a conditional expression that needs to retrieve data from another table, determine the number of hit selection rows for that table and add it to the number of updated rows.
- If conditions are specified for an index definition column, determine the combined total of the rows that are retrieved based only on the conditions for individual indexes. Furthermore, use the number of hit selection rows as the number of updated rows.

- Determine the number of hit selection rows after excluding conditions that use the columns of multiple tables and conditions that use functions. Furthermore, use the number of hit selection rows as the number of updated rows.
- Use the number of updated pages if a lock is specified for each page. If a unique index is defined for the table, also add the value for the rows to be updated.

[4] Repeat and add this for the number of LOB columns and LOB attributes.

[5] For the number of logical files used by each plug-in, refer to the manual provided with the plug-in.

(14) UPDATE (LOCK TABLE specified, index key value no locking used)

Condition	Number of lock requests
HiRDB/Single Server	[preprocessing][1] + *number of RDAREAs to be updated*[2] + 1 + Σ (*number of segments storing pre-update LOB data*)[3] Add the following if a data-type plug-in or index-type plug-in is defined for the columns to be updated: + *number of logical files used by plug-ins*[4] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if LOCK is specified for the pd_nowait_scan_option operand (add only for a non-FIX table): + 1 Add the following if routines are used for updating: + *number of routines used*
HiRDB/Parallel Server	

[1] For the number of lock requests in preprocessing, see (25).

[2] Total number of RDAREAs for the indexes, LOB columns, and LOB attributes.

[3] Repeat and add this for the number of LOB columns and LOB attributes.

[4] For the number of logical files used by each plug-in, refer to the manual provided with the plug-in.

(15) DELETE (LOCK TABLE not specified)

Condition		Number of lock requests
HiRDB/Single Server		[preprocessing][1] + *number of RDAREAs to be deleted*[5] + 1 + *number of deleted rows*[2] + *number of VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of branched rows* + *number of BINARY-type columns that cannot be stored in one page* × *number of branched rows* + *number of indexes* × *number of deleted rows* + Σ (*number of deleted LOB data segments* + *number of segments storing the deleted LOB data*)[4] Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by the plug-ins*[3] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for deletion: + *number of routines used*
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing][1] Add the following if routines are used for deletion: + *number of routines used*
	Back-end server	*number of RDAREAs to be deleted*[5] + 1 + *number of deleted rows*[2] + *number of VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of branched rows* + *number of BINARY-type columns that cannot be stored in one page* × *number of branched rows* + *number of indexes* × *number of deleted rows* + Σ (*number of deleted LOB data segments* + *number of segments storing the deleted LOB data*)[4] Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by the plug-ins*[3] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for deletion: + *number of routines used*

[1] For the number of lock requests in preprocessing, see (25).

[2]

- If there is a conditional expression that needs to retrieve data from another table, determine the number of hit selection rows for that table and add it to the number of deleted rows.
- If conditions are specified for an index definition column, determine the combined total of the rows that are retrieved based only on the conditions for individual indexes. Furthermore, use the number of hit selection rows as the number of deleted rows.

- Determine the number of hit selection rows after excluding conditions that use the columns of multiple tables and conditions that use functions. Furthermore, use the number of hit selection rows as the number of deleted rows.
- Use the number of deleted pages if a lock is specified for each page.

[3] For the number of logical files used by plug-ins, refer to the manuals included with the plug-ins.

[4] Repeat and add this for the number of LOB columns and LOB attributes.

[5] Total number of RDAREAs for the target tables, indexes, LOB columns, and LOB attributes.

(16) DELETE (LOCK TABLE specified[1])

Condition		Number of lock requests
HiRDB/Single Server		[preprocessing][2] + *number of RDAREAs to be deleted*[5] + Σ (*number of segments storing the deleted LOB data*)[4] + *number of released data pages*[6] Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by the plug-ins*[3] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for deletion: + *number of routines used*
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing][2] Add the following if routines are used for deletion: + *number of routines used*
	Back-end server	*number of RDAREAs to be deleted*[5] + Σ (*number of segments storing the deleted LOB data*[4] + *number of released data pages*[6] Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by the plug-ins*[3] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for deletion: + *number of routines used*

[1] If `LOCK TABLE IN SHARE MODE` is specified, the same formula should be used as when `LOCK TABLE` in not specified.

[2] For the number of lock requests in preprocessing, see (25).

[3] For the number of logical files used by plug-ins, refer to the manuals included with

the plug-ins.

[4] Repeat and add this for the number of LOB columns and LOB attributes.

[5] Total number of RDAREAs for the target indexes, LOB columns, and LOB attributes.

[6] Number of pages from which all data is deleted.

(17) DELETE (LOCK TABLE not specified, index key value no locking used)

Condition	Number of lock requests
HiRDB/Single Server	[preprocessing][1] + *number of RDAREAs to be deleted*[2] + 1 + *number of deleted rows*[3] + *number of VARCHAR, NVARCHAR, or MVARCHAR columns that are 256 bytes or larger* × *number of branched rows* + *number of BINARY-type columns that cannot be stored in one page* × *number of branched rows* + Σ (*number of deleted LOB data segments* + *number of segments storing the deleted LOB data*)[4] Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by plug-ins*[5] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for deletion: + *number of routines used*
HiRDB/Parallel Server	

[1] For the number of lock requests in preprocessing, see (25).

[2] Total number of RDAREAs for the target tables, indexes, LOB columns, and LOB attributes.

[3]

- If there is a conditional expression that needs to retrieve data from another table, determine the number of hit selection rows for that table and add it to the number of deleted rows.
- If conditions are specified for an index definition column, determine the combined total of the rows that are retrieved based only on the conditions for individual indexes. Furthermore, use the number of hit selection rows as the number of deleted rows.
- Determine the number of hit selection rows after excluding conditions that use the columns of multiple tables and conditions that use functions. Furthermore, use the number of hit selection rows as the number of deleted rows.

- Use the number of deleted pages if a lock is specified for each page. If a unique index is defined for the table, also add the value for the rows to be deleted.

[4] Repeat and add this for the number of LOB columns and LOB attributes.

[5] For the number of logical files used by each plug-in, refer to the manual provided with the plug-in.

(18) DELETE (LOCK TABLE specified, index key value no locking used)

Condition	Number of lock requests
HiRDB/Single Server	[preprocessing][1] + *number of RDAREAs to be deleted*[2] + Σ (*number of segments storing the deleted LOB data*)[3] + *number of released data pages*[4] Add the following if a data-type plug-in or index-type plug-in is defined: + *number of logical files used by plug-ins*[5] Add the following if the pd_inner_replica_control operand is specified: + 1 Add the following if routines are used for deletion: + *number of routines used*
HiRDB/Parallel Server	

[1] For the number of lock requests in preprocessing, see (25).

[2] Total number of RDAREAs for the target indexes, LOB columns, and LOB attributes.

[3] Repeat and add this for the number of LOB columns and LOB attributes.

[4] Number of pages from which all data is deleted.

[5] For the number of logical files used by each plug-in, refer to the manual provided with the plug-in.

(19) PURGE TABLE

Condition	Number of lock requests
HiRDB/Single Server	[preprocessing][1] + *number of target RDAREAs*[3] + 3 + *number of table segments being used* + Σ (*number of index segments being used* + 2)[2] + Σ (↑ *number of segments for which LOB data is stored in HiRDB file* ÷ 64000 ↑)[4] Add the following if the pd_inner_replica_control operand is specified: + 1

Condition		Number of lock requests
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing][1]
	Back-end server	*number of target RDAREAs*[3] + 3 + *number of table segments* + Σ (*number of index segments being used* + 2)[2] + Σ (↑ *number of segments for which LOB data is stored in HiRDB file* ÷ 64000 ↑)[4] Add the following if the pd_inner_replica_control operand is specified: + 1

[1] For the number of lock requests in preprocessing, see (25).

[2] Repeat and add this for the number of indexes.

[3] Total number of RDAREAs for the target tables, indexes, LOB columns, and LOB attributes.

[4] Repeat and add this for the number of HiRDB files for LOB RDAREAs of LOB columns and LOB attributes.

(20) ASSIGN LIST statement (list creation from a base table)

Condition		Number of lock requests
HiRDB/Single Server		*number of base table SELECT lock requests*[1] + 7 + 2 Add the following value if an existing list is specified as the name of the list to be created: + *number of RDAREAs for the base table of the list before re-creation* + 3 + *number of data segments of the list before re-creation*
HiRDB/Parallel Server	Front-end server	[preprocessing][2]
	Dictionary server	preprocessing[2] + 7 Add the following value if an existing list is specified as the name of the list to be created: + *number of RDAREAs for the base table of the list before re-creation*
	Back-end server	*number of base table SELECT lock requests*[1] + 2 Add the following value if an existing list is specified as the name of the list to be created: + 3 + *number of data segments of the list before re-creation*

[1] For the number of base table `SELECT` lock requests, choose from the `SELECT`

statements in (1) through (4) the statement that satisfies the condition of the base table and refer to it.

[2] For the number of lock requests in preprocessing, see (25).

(21) ASSIGN LIST statement (list creation from a list)

Condition		Number of lock requests
HiRDB/Single Server		[preprocessing]* + 7 + *number of RDAREAs storing the base table of the list* + 4 If an existing name is specified for the list to be created, add the following value: + *number of RDAREAs storing the base table of the list before re-creation* + 3 + *number of segments using the list data before re-creation*
HiRDB/Parallel Server	Front-end server	[preprocessing]*
	Dictionary server	[preprocessing]* + 7 If an existing name is specified for the list to be created, add the following value: + *number of RDAREAs storing the base table of the list before re-creation*
	Back-end server	*number of RDAREAs storing the base table of the list* + 4 If an existing name is specified for the list to be created, add the following value: + 3 + *number of segments using the list data before re-creation*

* For the number of lock requests in preprocessing, see (25).

(22) ASSIGN LIST statement (list name change)

Condition	Number of lock requests
HiRDB/Single Server	7 If an existing name is specified for the list to be created, add the following value: + *number of RDAREAs storing the base table of the list before re-creation*
HiRDB/Parallel Server (Dictionary server)	

(23) DROP LIST statement (deletion with list name specification: DROP LIST ～)

Condition	Number of lock requests
HiRDB/Single Server	7 + *number of RDAREAs storing the base table of the list* + 3 + *number of segments using the list data*

Condition		Number of lock requests
HiRDB/Parallel Server	Dictionary server	7 + *number of RDAREAs storing the base table of the list*
	Back-end server	3 + *number of segments using the list data*

(24) DROP LIST statement (deletion of all lists locally owned: DROP ALL LIST ～)

Condition		Number of lock requests
HiRDB/Single Server		1 + *number of RDAREAs for lists* + *number of deleted lists* × 2 + *total number of segments using the deleted list data*
HiRDB/Parallel Server	Dictionary server	1
	Back-end server	*Number of RDAREAs for lists* + *number of deleted lists* × 2 + *total number of segments using the deleted list data*

(25) Number of lock requests in preprocessing

Condition	Number of lock requests
HiRDB/Single Server	1 + *number of tables used* + *number of view tables used* + Σ^{1}(2 + *number of columns* × 2 + *number of indexes* × 2 + *number of table partitions* × 4 + Σ^{2}(*number of partitioning indexes* × 2) + 2) + 13 Add the following if a user-defined type or function is used: + 1 Add the following (for each table) if optimization information is collected: + 1 + *number of columns for which optimization information is created* × 4 + 1 Add the following (for each view table) if view tables are used: + Σ (*number of columns* × 2 + 4 + *number of columns comprising the view table* + 2) Add the following if LOB columns are defined: + *number of LOB columns* × 2 + *number of table partitions* Add the following if user-defined type columns are defined: *number of user-defined type columns* + Σ (4 + *number of attributes* × 2) Add the following if LOB attributes are used: + *number of LOB attributes* × 2 + *number of table partitions* Add the following if a higher-order type is used: + 2 Add the following for each function used if functions are used: + Σ (2 + *number of functions with the same name and parameter configuration* × 2 + *number of parameters* × 2) Add the following for each plug-in used if plug-ins are used: + Σ (2 + *number of parameters* × 2) + 3 + *number of contexts* × 4 + 2 Add the following if routines are used: + *number of routines used*

Condition		Number of lock requests
HiRDB/Parallel Server	Front-end server	1 + *number of tables used* + *number of view tables used* Add the following if a user-defined type or function is used: + 1 Add the following if routines are used: + *number of routines used*
	Dictionary server	Σ[1](2 + *number of columns* × 2 + *number of indexes* × 2 + *number of table partitions* × 4 + Σ[2](*number of partitioning indexes* × 2) + 2) + 13 Add the following (for each table) if optimization information is collected: + 1 + *number of columns for which optimization information is created* × 4 + 1 Add the following (for each view table) if view tables are used: + Σ (*number of columns* × 2 + 4 + *number of columns comprising the view table* + 2) Add the following if LOB columns are defined: + *number of LOB columns* × 2 + *number of table partitions* Add the following for each user-defined type column if user-defined type columns are defined: + Σ (4 + *number of attributes* × 2) Add the following if LOB attributes are used: + *number of LOB attributes* × 2 + *number of table partitions* Add the following if a higher-order type is used: + 2 Add the following for each function used if functions are used: + Σ (2 + *number of functions with the same name and parameter configuration* × 2 + *number of parameters* × 2) Add the following for each plug-in used if plug-ins are used: + Σ (2 + *number of parameters* × 2) + 3 + *number of contexts* × 4 + 2 Add the following if routines are used: + *number of routines used*

[1] Summarize this for each table.

[2] Summarize this for each partitioning index inside the table.

E.3 Control SQL

(1) LOCK

Condition	Number of lock requests
HiRDB/Single Server	[preprocessing]* + *number of table row partitions* + 1

Condition		Number of lock requests
HiRDB/Parallel Server	Front-end server and Dictionary server	[preprocessing]*
	Back-end server	*Number of table row partitions* + 1

* For the number of lock requests in preprocessing, see in Section *E.2(25) Number of lock requests in preprocessing*.

E.4 Utilities and commands

(1) Database load utility (pdload)

Condition		Number of lock requests
HiRDB/Single Server		209 + α + *number of segments used by RDAREAs for table* + *number of segments used by RDAREAs for index* + (↑ *number of segments used by LOB RDAREAs* ÷ 64000 ↑) + β
HiRDB/Parallel Server	Front-end server	3
	Dictionary server	206 + α
	Back-end server	*Number of segments used by RDAREAs for table* + *number of segments used by RDAREAs for index* + (↑ *number of segments used by LOB RDAREAs* ÷ 64000 ↑) + β

α : MAX(*number of table columns*, *number of table indexes*, *number of RDAREAs for table*)

β : To estimate a value, see the section on *Resources to be used in locking across transactions (specified using the pd_lck_until_disconnect_cnt operand)* in *Locked resources per server that are required for executing the database reorganization utility* in the manual *HiRDB Version 8 Command Reference*.

Determine the number of segments used from the number of segments scheduled to store data. When the `-d` option is specified and if the number of segments already storing data (the number of segments being used) is greater than the number of segments scheduled to store data, use for computation the number of segments being used.

(2) Database reorganization utility (pdrorg)

Condition		Number of lock requests
HiRDB/Single Server		209 + α + *number of segments used by RDAREAs* for table + *number of segments used by RDAREAs for index* (↑ *number of segments used by LOB RDAREAs* ÷ 64000 ↑) + β
HiRDB/Parallel Server	Front-end server	3
	Dictionary server	206 + α
	Back-end server	*Number of segments used by RDAREAs for table* + *number of segments used by RDAREAs for index* + (↑ *number of segments used by LOB RDAREAs* ÷ 64000 ↑) + β

α : MAX(*number of table columns*, *number of table indexes*, *number of RDAREAs for table*)

β : To estimate a value, see the section on *Resources to be used in locking across transactions (specified using the pd_lck_until_disconnect_cnt operand)* in *Locked resources per server that are required for executing the database reorganization utility* in the manual *HiRDB Version 8 Command Reference*.

(3) Rebalancing utility (pdrbal)

Condition			Number of lock requests
HiRDB/ Single Server	Shared mode (`-k share`)		starting/finishing[1] + preprocessing[2] + rebalancing[3] + 2
	Exclusive mode (`-k exclusive`)		starting/finishing[1] + preprocessing[2] + rebalancing[3]
HiRDB/ Parallel Server	Shared mode (`-k share`)	Front-end server	starting/finishing[1] + preprocessing[2] + 1
		Dictionary server	starting/finishing[1] + preprocessing[2]
		Back-end server	starting/finishing[1] + rebalancing[3]
	Exclusive mode (`-k exclusive`)	Front-end server	starting/finishing[1] + preprocessing[2]
		Dictionary server	starting/finishing[1] + preprocessing[2]
		Back-end server	starting/finishing[1] + rebalancing[3]

[1] Number of lock requests for starting/finishing = $\Sigma A_i + \Sigma B_i$

A_i: Number of lock requests per function

For the number of lock requests of a function, see in *E.1(5) CREATE FUNCTION*.

B_i: Number of lock requests per procedure

For the number of lock requests of a procedure, see in *E.1(8) CREATE PROCEDURE*.

[2] The number of lock requests for preprocessing is as follows:

- HiRDB/Single Server: 225
- HiRDB/Parallel Server (front-end server): 5
- HiRDB/Parallel Server (dictionary server): 220

[3] The number of lock requests for rebalancing = *number of segments used by RDAREAs for table* + *number of segments used by RDAREAs for index* + (↑ *number of segments used by LOB RDAREAs* ÷ 64000 ↑)

(4) pdacunlck command

Condition	Number of lock requests
HiRDB/Single Server	0
HiRDB/Parallel Server (dictionary server)	Add the following if an authorization identifier is specified: + 128 Add the following if ALL is specified: + *number-of-users-registered-in-dictionary-table-SQL_USERS*

F. Operands Checked by the pdconfchk Command

Table F-1 shows the operands that are checked by the pdconfchk command (the operands are listed in alphabetical order). Note that the operands of the following definitions are not checked by the pdconfchk command.

- UAP environment definition
- Foreign server information definition (HiRDB External Data Access facility)
- Hub optimization information definition (HiRDB External Data Access facility)

Table F-1: Operands checked by the pdconfchk command

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
HORCMINST	Y	N	N	N	N	Y
pd_additional_optimize_level	Y	N	N	N	N	Y
pd_alias_cache_size	Y	N	N	N	N	Y
pd_assurance_index_no	Y	N	N	N	N	Y
pd_assurance_table_no	Y	N	N	N	N	Y
pd_audit	Y	N	N	N	N	Y
pd_aud_async_buff_count	Y	N	N	N	N	Y
pd_aud_async_buff_retry_intvl	Y	N	N	N	N	N
pd_aud_async_buff_size	Y	N	N	N	N	Y
pd_aud_file_name	Y	N	N	N	N	Y
pd_aud_file_wrn_pnt	Y	N	N	N	N	Y
pd_aud_max_generation_num	Y	N	N	N	N	Y
pd_aud_max_generation_size	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_aud_no_standby_file_opr	Y	N	N	N	N	Y
pd_audit_def_buffer_size	Y	N	N	N	N	N
pd_auth_cache_size	Y	N	N	N	N	Y
pd_auto_vrup	Y	N	N	N	N	Y
pd_bes_connection_hold	Y	N	N	N	N	N
pd_bes_conn_hold_trn_interval	Y	N	N	N	N	N
pd_bes_shmpool_size	Y	N	N	N	N	Y
pd_cancel_down_msgchange	Y	N	N	N	N	Y
pd_cancel_dump	Y	N	N	N	N	Y
pd_change_clt_ipaddr	Y	N	N	N	N	Y
pd_check_pending	Y	N	N	N	N	N
pd_client_waittime_over_abort	Y	N	N	N	N	Y
pd_cmdhold_precheck	Y	N	N	N	N	Y
pd_command_deadlock_priority	Y	N	N	N	N	Y
pd_connect_errmsg_hide	Y	N	N	N	N	Y
pd_constraint_name	Y	N	N	N	N	N
pd_cwaittime_report_dir	Y	N	N	N	N	N
pd_cwaittime_report_size	Y	N	N	N	N	N
pd_cwaittime_wrn_pnt	Y	N	N	N	N	N
pd_dbbuff_attribute	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_dbbuff_lock_interval	Y	N	N	N	N	Y
pd_dbbuff_lock_release_detect	Y	N	N	N	N	Y
pd_dbbuff_lock_spn_count	Y	N	N	N	N	Y
pd_dbbuff_lru_option	Y	N	N	N	N	Y
pd_dbbuff_modify	Y	N	N	N	N	Y
pd_dbbuff_rate_updpage	Y	N	N	N	N	Y
pd_dbbuff_wait_interval	Y	N	N	N	N	Y
pd_dbbuff_wait_spn_count	Y	N	N	N	N	Y
pd_dbsync_altwrite_skip	Y	N	N	N	N	Y
pd_dbsync_lck_release_count	Y	N	N	N	N	Y
pd_dbsync_point	Y	N	N	N	N	Y
pd_db_io_error_action	Y	N	N	N	N	N
pd_deadlock_priority_use	Y	N	N	N	N	Y
pd_debug_info_netstat	Y	N	N	N	N	Y
pd_dec_sign_normalize	Y	N	N	N	N	Y
pd_def_buf_control_area_assign	N	N	N	N	N	N
pd_delete_reserved_word_file	Y	N	N	N	N	Y
pd_dfw_awt_process	Y	N	N	N	N	Y
pd_dfw_syncpoint_skip_limit	Y	N	N	N	N	N

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_dic_shmpool_size	Y	N	N	N	N	Y
pd_directory_server	Y	N	N	N	N	Y
pd_down_watch_proc	Y	N	N	N	N	Y
pd_dump_suppress_watch_time	Y	N	N	N	N	Y
pd_fes_lck_pool_size	Y	N	N	N	N	Y
pd_floatable_bes	Y	N	N	N	N	N
pd_foreign_server_libpath	Y	N	N	N	N	N
pd_ha	Y	N	N	N	N	Y
pd_ha_acttype	Y	N	N	N	N	N
pd_ha_agent	Y	N	N	N	N	N
pd_ha_ipaddr_inherit	Y	N	N	N	N	Y
pd_ha_max_act_guest_servers	Y	N	N	N	N	Y
pd_ha_max_server_process	Y	N	N	N	N	Y
pd_ha_mgr_rerun	Y	N	N	N	N	Y
pd_ha_prc_cleanup_check	Y	N	N	N	N	Y
pd_ha_process_count	Y	N	N	N	N	Y
pd_ha_resource_act_wait_time	Y	N	N	N	N	Y
pd_ha_restart_failure	Y	N	N	N	N	N
pd_ha_server_process_standby	Y	N	N	N	N	N
pd_ha_switch_timeout	Y	N	N	N	N	Y
pd_ha_transaction	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_ha_trn_queuing_wait_time	Y	N	N	N	N	Y
pd_ha_trn_restart_retry_time	Y	N	N	N	N	Y
pd_ha_unit	Y	N	N	N	N	N
pd_hash_table_size	Y	N	N	N	N	Y
pd_hashjoin_hashing_mode	Y	N	N	N	N	Y
pd_hostname	Y	N	N	N	N	Y
pd_host_watch_interval	Y	N	N	N	N	Y
pd_indexlock_mode	Y	N	N	N	N	Y
pd_inner_replica_control	Y	N	N	N	N	N
pd_ipc_conn_count	Y	N	N	N	N	Y
pd_ipc_conn_interval	Y	N	N	N	N	Y
pd_ipc_conn_nblock	Y	N	N	N	N	Y
pd_ipc_conn_nblock_time	Y	N	N	N	N	Y
pd_ipc_inet_bufsize	Y	N	N	N	N	Y
pd_ipc_recv_count	Y	N	N	N	N	Y
pd_ipc_send_count	Y	N	N	N	N	Y
pd_ipc_send_retrycount	Y	N	N	N	N	Y
pd_ipc_send_retrysleeptime	Y	N	N	N	N	Y
pd_ipc_tcp_nodelayack	Y	N	N	N	N	Y
pd_ipc_unix_bufsize	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_java_archive_directory	Y	N	N	N	N	Y
pd_java_classpath	Y	N	N	N	N	Y
pd_java_libpath	Y	N	N	N	N	Y
pd_java_option	Y	N	N	N	N	Y
pd_java_routine_stack_size	Y	N	N	N	N	Y
pd_java_runtimepath	Y	N	N	N	N	Y
pd_java_stdout_file	Y	N	N	N	N	Y
pd_jp1_event_level	Y	N	N	N	N	Y
pd_jp1_event_msg_out	Y	N	N	N	N	Y
pd_jp1_use	Y	N	N	N	N	Y
pd_key_resource_type	Y	N	N	N	N	Y
pd_large_file_use	Y	N	N	N	N	Y
pd_lck_deadlock_info	Y	N	N	N	N	Y
pd_lck_hash_entry	Y	N	N	N	N	Y
pd_lck_pool_size	Y	N	N	N	N	Y
pd_lck_queue_limit	Y	N	N	N	N	Y
pd_lck_release_detect	Y	N	N	N	N	Y
pd_lck_release_detect_interval	Y	N	N	N	N	Y
pd_lck_until_disconnect_cnt	Y	N	N	N	N	Y
pd_lck_wait_timeout	Y	N	N	N	N	Y
pd_list_initialize_timing	Y	N	N	N	N	N
pd_log_auto_unload_path	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_log_dual	Y	N	N	N	N	Y
pd_log_dual_write_method	Y	N	N	N	N	N
pd_log_max_data_size	Y	N	N	N	N	Y
pd_log_org_no_standby_file_opr	Y	N	N	N	N	N
pd_log_org_reflected_logpoint	Y	N	N	N	N	N
pd_log_rec_leng	Y	N	N	N	N	Y
pd_log_remain_space_check	Y	N	N	N	N	N
pd_log_rerun_reserved_file_open	Y	N	N	N	N	Y
pd_log_rerun_swap	Y	N	N	N	N	Y
pd_log_rpl_no_standby_file_opr	Y	N	N	N	N	Y
pd_log_sdinterval	Y	N	N	N	N	Y
pd_log_singleoperation	Y	N	N	N	N	Y
pd_log_swap_timeout	Y	N	N	N	N	Y
pd_log_unload_check	Y	N	N	N	N	Y
pd_log_write_buff_count	Y	N	N	N	N	Y
pd_lv_mirror_use	Y	N	N	N	N	N
pd_master_file_name	Y	N	N	N	N	Y
pd_max_access_tables	Y	N	N	N	N	Y
pd_max_access_tables_wrn_pnt	Y	N	N	N	N	Y
pd_max_add_dbbuff_no	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_max_add_dbbuff_shm_no	Y	N	N	N	N	Y
pd_max_ard_process	Y	N	N	N	N	N
pd_max_bes_process	Y	N	N	N	N	Y
pd_max_commit_write_reclaim_no	Y	N	N	N	N	Y
pd_max_dic_process	Y	N	N	N	N	Y
pd_max_file_no	Y	N	N	N	N	Y
pd_max_file_no_wrn_pnt	Y	N	N	N	N	Y
pd_max_foreign_server	Y	N	N	N	N	Y
pd_max_list_users	Y	N	N	N	N	Y
pd_max_list_count	Y	N	N	N	N	Y
pd_max_list_users_wrn_pnt	Y	N	N	N	N	Y
pd_max_list_count_wrn_pnt	Y	N	N	N	N	Y
pd_max_open_fds	Y	N	N	N	N	Y
pd_max_open_holdable_cursors	Y	N	N	N	N	Y
pd_max_rdarea_no	Y	N	N	N	N	Y
pd_max_rdarea_no_wrn_pnt	Y	N	N	N	N	Y
pd_max_recover_process	Y	N	N	N	N	Y
pd_max_reflect_process_count	Y	N	N	N	N	Y
pd_max_server_process	Y	N	N	N	N	Y
pd_max_users	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_max_users_wrn_pnt	Y	N	N	N	N	Y
pd_mlg_file_size	Y	N	N	N	N	Y
pd_mlg_msg_log_unit	Y	N	N	N	N	Y
pd_mode_conf	Y	N	N	N	N	Y
pd_module_trace_max	Y	N	N	N	N	Y
pd_module_trace_timer_level	Y	N	N	N	N	Y
pd_name_port	Y	N	N	N	N	Y
pd_node_name	Y	N	N	N	N	N
pd_non_floatable_bes	Y	N	N	N	N	N
pd_nowait_scan_option	Y	N	N	N	N	Y
pd_oltp_holdcr	Y	N	N	N	N	Y
pd_optimize_level	Y	N	N	N	N	Y
pd_overflow_suppress	Y	N	N	N	N	Y
pd_pageaccess_mode	Y	N	N	N	N	Y
pd_plugin_ixmk_dir	Y	N	N	N	N	N
pd_process_count	Y	N	N	N	N	Y
pd_process_terminator	Y	N	N	N	N	Y
pd_process_terminator_max	Y	N	N	N	N	Y
pd_queue_watch_time	Y	N	N	N	N	Y
pd_queue_watch_timeover_action	Y	N	N	N	N	Y
pd_rdarea_list_no_wrn_pnt	Y	N	N	N	N	Y
pd_rdarea_open_attribute	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_rdarea_open_attribute_use	Y	N	N	N	N	Y
pd_rdarea_warning_point	Y	N	N	N	N	Y
pd_redo_allpage_put	Y	N	N	N	N	Y
pd_reduced_check_time	Y	N	N	N	N	Y
pd_registered_port	Y	Y	Y	Y	Y	Y
pd_registered_port_check	Y	Y	Y	Y	Y	Y
pd_registered_port_level	Y	Y	Y	Y	Y	Y
pd_registry_cache_size	Y	N	N	N	N	Y
pd_rise_disaster_mode	Y	N	N	N	N	Y
pd_rise_fence_level	Y	N	N	N	N	Y
pd_rise_pairvolume_combination	Y	N	N	N	N	Y
pd_rise_use	Y	N	N	N	N	Y
pd_rorg_predict	Y	N	N	N	N	Y
pd_routine_def_cache_size	Y	N	N	N	N	Y
pd_rpc_trace	Y	N	N	N	N	Y
pd_rpc_trace_name	Y	N	N	N	N	Y
pd_rpc_trace_size	Y	N	N	N	N	Y
pd_rpl_hdepath	Y	N	N	N	N	N
pd_rpl_init_start	Y	N	N	N	N	Y
pd_rpl_reflect_mode	Y	N	N	N	N	Y
pd_sds_shmpool_size	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_server_cleanup_interval	Y	N	N	N	N	Y
pd_server_entry_queue	Y	N	N	N	N	Y
pd_service_port	Y	N	N	N	N	Y
pd_shared_rdarea_use	Y	N	N	N	N	Y
pd_shmpool_attribute	Y	N	N	N	N	Y
pd_space_level	Y	N	N	N	N	Y
pd_spd_assurance_count	Y	N	N	N	N	Y
pd_spd_assurance_msg	Y	N	N	N	N	Y
pd_spd_dual	Y	N	N	N	N	Y
pd_spd_max_data_size	Y	N	N	N	N	Y
pd_spd_reduced_mode	Y	N	N	N	N	Y
pd_spd_reserved_file_auto_open	Y	N	N	N	N	Y
pd_spd_syncpoint_skip_limit	Y	N	N	N	N	Y
pd_spool_cleanup	Y	N	N	N	N	N
pd_spool_cleanup_interval	Y	N	N	N	N	N
pd_spool_cleanup_interval_level	Y	N	N	N	N	N
pd_spool_cleanup_level	Y	N	N	N	N	N
pd_sql_object_cache_size	Y	N	N	N	N	Y
pd_sql_send_buff_size	Y	N	N	N	N	Y
pd_start_level	Y	N	N	N	N	Y
pd_start_skip_unit	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_start_time_out	Y	N	N	N	N	Y
pd_statistics	Y	N	N	N	N	Y
pd_stj_file_size	Y	N	N	N	N	Y
pd_stj_buff_size	Y	N	N	N	N	Y
pd_sts_file_name_1-7	Y	Y	Y	Y	N	N
pd_sts_initial_error	Y	N	N	N	N	Y
pd_sts_last_active_file	Y	N	N	N	N	N
pd_sts_last_active_side	Y	N	N	N	N	N
pd_sts_last_active_side_sub	Y	N	N	N	N	N
pd_sts_last_active_subfile	Y	N	N	N	N	N
pd_sts_singleoperation	Y	N	N	N	N	Y
pd_sts_subfile_name_1 to 7	Y	Y	Y	Y	N	N
pd_substr_length	Y	N	N	N	N	Y
pd_svr_castoff_size	Y	N	N	N	N	Y
pd_sysdef_default_option	Y	N	N	N	N	Y
pd_syssts_file_name_1-7	Y	N	N	N	N	Y
pd_syssts_initial_error	Y	N	N	N	N	Y
pd_syssts_last_active_file	Y	N	N	N	N	Y
pd_syssts_last_active_side	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_syssts_last_active_side_sub	Y	N	N	N	N	N
pd_syssts_last_active_subfile	Y	N	N	N	N	N
pd_syssts_singleoperation	Y	N	N	N	N	Y
pd_syssts_subfile_name_1 to 7	Y	Y	Y	Y	N	N
pd_system_complete_wait_time	Y	N	N	N	N	Y
pd_system_dbsync_point	Y	N	N	N	N	Y
pd_system_id	Y	N	N	N	N	Y
pd_table_def_cache_size	Y	N	N	N	N	Y
pd_tcp_inet_bufsize	Y	N	N	N	N	Y
pd_tcp_unix_bufsize	Y	N	N	N	N	Y
pd_term_watch_count	Y	N	N	N	N	Y
pd_thdlock_pipe_retry_interval	Y	N	N	N	N	Y
pd_thdlock_retry_time	Y	N	N	N	N	Y
pd_thdlock_sleep_func	Y	N	N	N	N	Y
pd_thdlock_wakeup_lock	Y	N	N	N	N	Y
pd_thdspnlk_spn_count	Y	N	N	N	N	Y
pd_thred_max_stack_size	Y	N	N	N	N	Y
pd_trn_rerun_branch_auto_decide	Y	N	N	N	N	Y
pd_trn_commit_optimize	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_trn_send_decision_interval	Y	N	N	N	N	Y
pd_trn_send_decision_intval_sec	Y	N	N	N	N	Y
pd_trn_send_decision_retry_time	Y	N	N	N	N	Y
pd_trn_watch_time	Y	N	N	N	N	Y
pd_type_def_cache_size	Y	N	N	N	N	Y
pd_uap_exerror_log_dir	Y	N	N	N	N	Y
pd_uap_exerror_log_param_size	Y	N	N	N	N	Y
pd_uap_exerror_log_size	Y	N	N	N	N	Y
pd_uap_exerror_log_use	Y	N	N	N	N	Y
pd_unit_id	Y	N	N	N	N	Y
pd_utl_buff_size	Y	N	N	N	N	Y
pd_utl_exec_mode	Y	N	N	N	N	Y
pd_utl_exec_time	Y	N	N	N	N	Y
pd_view_def_cache_size	Y	N	N	N	N	Y
pd_watch_pc_client_time	Y	N	N	N	N	Y
pd_watch_resource	Y	N	N	N	N	Y
pd_watch_time	Y	N	N	N	N	Y
pd_work_buff_expand_limit	Y	N	N	N	N	Y
pd_work_buff_mode	Y	N	N	N	N	Y

Operand	Syntax check	File check	Access privilege check	Duplicate specification check	Host name check	Check among server machines
pd_work_buff_size	Y	N	N	N	N	Y
pd_work_table_option	Y	N	N	N	N	Y
pdwork_wrn_pnt	Y	N	N	N	N	Y
pdbuffer	Y	N	N	N	N	Y
pdcltgrp	Y	N	N	N	N	Y
PDCONFPATH	Y	N	N	N	N	Y
pdhagroup	Y	N	N	Y	N	Y
pdhibegin	Y	N	N	N	N	Y
pdhubopt	Y	N	N	N	N	N
pdlogadfg -d spd	Y	Y	Y	Y	N	N
pdlogadfg -d ssp	Y	Y	Y	Y	N	N
pdlogadfg -d sys	Y	Y	Y	Y	N	N
pdlogadpf -d spd	Y	Y	Y	Y	N	N
pdlogadpf -d ssp	Y	Y	Y	Y	N	N
pdlogadpf -d sys	Y	Y	Y	Y	N	N
pdmlgput	Y	N	N	N	N	Y
pdplgprm	Y	N	N	N	N	N
pdplugin	Y	N	N	N	N	Y
pdstart	Y	N	N	N	Y	Y
pdstbegin	Y	N	N	N	N	Y
pdunit	Y	N	N	N	Y	Y
pdwork	Y	N	N	N	N	N
SHMMAX	Y	N	N	N	N	Y
TZ	Y	N	N	N	N	Y

Y: Operand is checked.

N: Operand cannot be checked.

Syntax check:

Checks whether or not the operand's syntax is correct.

File check:

Checks for the presence of the system log file, synchronization point dump file, and status file. File check is not performed if the `-n` option is specified for the `pdconfchk` command.

Access privilege check:

Checks whether or not the HiRDB administrator has file access privileges.

Checks whether or not the HiRDB administrator has hosts file access privilege.

Access privilege check is not performed if the `-n` option is specified for the `pdconfchk` command.

Duplicate specification check:

Checks whether or not the system log files, synchronization point dump files, or status files are duplicated.

Host name check:

Checks whether or not the host name is described in the hosts file.

Check among server machines (applicable only to HiRDB/Parallel Server):

Checks for compatibility among server machines using the server machine of the system manager as the reference.

G. Operands Created by the pdgen Command

Table G-1 shows the operands created by the pdgen command. Operands not listed in this table are not created by this command. The operand names are listed alphabetically. Note that the operands for the following definitions are not created by the pdgen command.

- UAP environment definition
- Foreign server information definition (HiRDB External Data Access facility)
- Hub optimization information definition (HiRDB External Data Access facility)

Table G-1: Operands created by the pdgen command

Operand name	Explanation
pd_additional_optimize_level	COST_BASE_2 is set.
pd_alias_cache_size	0 is used.
pd_auth_cache_size	The value specified by pdgen is used.
pd_dbbuff_lru_option	These operands are output as comment statements.
pd_dbsync_point	
pd_deadlock_priority_use	
pd_ha	nouse is used.
pd_ha_ipaddr_inherit	N is used.
pd_ha_restart_failure	These operands are output as comment statements.
pd_ha_unit	
pd_hostname	The value determined from the specification value of pdgen is used.
pd_host_watch_interval	10 is used.
pd_jp1_use	N is used.
pd_large_file_use	Y is used.
pd_lck_deadlock_info	
pd_lck_pool_size	The value specified by pdgen is used. If pdgen is not specified, this operand is not generated.
pd_lck_release_detect	pipe is used.

Operand name	Explanation
pd_lck_until_disconnect_cnt	256 is used.
pd_lck_wait_timeout	The value specified by pdgen is used. If pdgen is not specified, this operand is not generated.
pd_log_dual	The value specified by pdgen is used.
pd_log_max_data_size	32,000 is used.
pd_log_rec_leng	1,024 is used.
pd_log_remain_space_check	safe is used.
pd_log_rerun_reserved_file_open	Y is used.
pd_log_rerun_swap	N is used.
pd_log_sdinterval	The value specified by pdgen is used.
pd_log_singleoperation	N is used.
pd_log_unload_check	The value specified by pdgen is used. Note that N is used for the pd_log_unload_check operand of a front-end server definition.
pd_log_write_buff_count	3 is used.
pd_master_file_name	The value determined from the specification value of pdgen is used.
pd_max_access_tables	The value specified by pdgen is used. If pdgen is not specified, this operand is not generated.
pd_max_bes_process	These operands are output as comment statements.
pd_max_dic_process	
pd_max_file_no	The value specified by pdgen is used. If pdgen is not specified, this operand is not generated.
pd_max_list_users	The value specified by pdgen is used.
pd_max_list_count	
pd_max_rdarea_no	The value specified by pdgen is used. If pdgen is not specified, this operand is not generated.
pd_max_recover_process	1 or 3 is used.
pd_max_users	The value specified by pdgen is used.
pd_mlg_file_size	1,024 is used.

Operand name	Explanation
pd_mode_conf	AUTO is used.
pd_name_port	The value specified by pdgen is used.
pd_optimize_level	■ HiRDB/Single Server: PRIOR_NEST_JOIN, PRIOR_OR_INDEXES, DETER_AND_INDEXES, RAPID_GROUPING, DETER_WORK_TABLE_FOR_UPDATE, and APPLY_ENHANCED_KEY_COND are used. ■ HiRDB/Parallel Server: PRIOR_NEST_JOIN, PRIOR_OR_INDEXES, SORT_DATA_BES, DETER_AND_INDEXES, RAPID_GROUPING, DETER_WORK_TABLE_FOR_UPDATE, and APPLY_ENHANCED_KEY_COND are used.
pd_overflow_suppress	N is used.
pd_process_count	The value specified by pdgen is used.
pd_rdarea_warning_point	These operands are output as comment statements.
pd_registry_cache_size	
pd_routine_def_cache_size	The value specified by pdgen is used.
pd_rpc_trace	N is used.
pd_rpc_trace_name	1,000,000 is used.
pd_rpc_trace_size	/tmp is used.
pd_shmpool_attribute	fixed is used.
pd_spd_assurance_count	1 is used.
pd_spd_dual	A value is set according to the number of HiRDB file system areas in effect when pdgen is executed.
pd_spd_max_data_size	32,000 is used.
pd_sql_object_cache_size	The value specified by pdgen is used.
pd_statistics	N is used.
pd_stj_buff_size	128 is used.
pd_sts_file_name_1-7	The value determined from the specification value of pdgen is used.
pd_sts_initial_error	stop is used.

Operand name	Explanation
pd_sts_last_active_file	These operands are output as comment statements.
pd_sts_last_active_side	
pd_sts_last_active_side_sub	
pd_sts_last_active_subfile	
pd_sts_singleoperation	stop is used.
pd_sts_subfile_name_1 to 7	The value determined from the specification value of pdgen is used.
pd_syssts_file_name_1 to 7	
pd_syssts_initial_error	stop is used.
pd_syssts_last_active_file	These operands are output as comment statements.
pd_syssts_last_active_side	
pd_syssts_last_active_side_sub	
pd_syssts_last_active_subfile	
pd_syssts_singleoperation	stop is used.
pd_syssts_subfile_name_1 to 7	The value determined from the specification value of pdgen is used.
pd_system_id	The value specified by pdgen is used.
pd_table_def_cache_size	
pd_type_def_cache_size	
pd_unit_id	SDSU is used for a HiRDB/Single Server. u*XXX* is used for a HiRDB/Parallel Server (where *XXX* is a number).
pd_utl_buff_size	32 is used.
pd_view_def_cache_size	The value specified by pdgen is used.
pd_watch_pc_client_time	This operand is output as a comment statement.
pd_watch_time	3,600 is used.
pd_work_buff_size	The value specified by pdgen is used.
pdbuffer	The value determined from the specification value of pdgen is used.
PDCONFPATH	This operand is output as a comment statement.

Operand name	Explanation
pdlogadfg -d spd	The value determined from the specification value of pdgen is used.
pdlogadfg -d ssp	
pdlogadfg -d sys	
pdlogadpf -d spd	
pdlogadpf -d ssp	
pdlogadpf -d sys	
pdplugin	The value specified by pdgen is used.
pdstart	The value determined from the specification value of pdgen is used.
pdunit	
pdwork	
SHMMAX	These operands are output as comment statements.
TZ	

H. Operands Specifiable Under Each OS

Table H-1 shows the operands that can be specified for each OS (the operands are listed in alphabetical order). Note that the operands of the following definitions are not included in table H-1.

- Foreign server information definition (HiRDB External Data Access facility)
- Hub optimization information definition (HiRDB External Data Access facility)

The operands for the foreign server information definition and Hub optimization information definition are applicable only to the HP-UX and AIX 5L versions. Additionally, both of the following conditions must be satisfied:

- POSIX library version of HiRDB
- 32-bit mode HiRDB

Table H-1: Operands Specifiable under each OS

Operand	HP-UX	Solaris	AIX 5L	Linux
HORCMINST	Y	N	Y	N
pd_additional_optimize_level	Y	Y	Y	Y
pd_alias_cache_size	Y	Y	Y	Y
pd_assurance_index_no	Y	Y	Y	Y
pd_assurance_table_no	Y	Y	Y	Y
pd_audit	Y	Y	Y	Y
pd_aud_async_buff_count	Y	Y	Y	Y
pd_aud_async_buff_retry_intvl	Y	Y	Y	Y
pd_aud_async_buff_size	Y	Y	Y	Y
pd_aud_file_name	Y	Y	Y	Y
pd_aud_file_wrn_pnt	Y	Y	Y	Y
pd_aud_max_generation_num	Y	Y	Y	Y
pd_aud_max_generation_size	Y	Y	Y	Y
pd_aud_no_standby_file_opr	Y	Y	Y	Y
pd_audit_def_buffer_size	Y	Y	Y	Y

Operand	HP-UX	Solaris	AIX 5L	Linux
pd_auth_cache_size	Y	Y	Y	Y
pd_auto_vrup	Y	Y	Y	Y
pd_bes_connection_hold	Y	Y	Y	Y
pd_bes_conn_hold_trn_interval	Y	Y	Y	Y
pd_bes_shmpool_size	Y	Y	Y	Y
pd_cancel_down_msgchange	Y	Y	Y	Y
pd_cancel_dump	Y	Y	Y	Y
pd_change_clt_ipaddr	Y	Y	Y	Y
pd_check_pending	Y	Y	Y	Y
pd_client_waittime_over_abort	Y	Y	Y	Y
pd_cmdhold_precheck	Y	Y	Y	Y
pd_command_deadlock_priority	Y	Y	Y	Y
pd_connect_errmsg_hide	Y	Y	Y	Y
pd_constraint_name	Y	Y	Y	Y
pd_cwaittime_report_dir	Y	Y	Y	Y
pd_cwaittime_report_size	Y	Y	Y	Y
pd_cwaittime_wrn_pnt	Y	Y	Y	Y
pd_dbbuff_attribute	Y	Y	C	N
pd_dbbuff_lock_interval	Y	Y	Y	Y
pd_dbbuff_lock_release_detect	Y	Y	Y	Y
pd_dbbuff_lock_spn_count	Y	Y	Y	Y
pd_dbbuff_lru_option	Y	Y	Y	Y
pd_dbbuff_modify	Y	Y	Y	Y
pd_dbbuff_rate_updpage	Y	Y	Y	Y
pd_dbbuff_wait_interval	Y	Y	Y	Y
pd_dbbuff_wait_spn_count	Y	Y	Y	Y
pd_dbsync_altwrite_skip	Y	Y	Y	Y

Operand	HP-UX	Solaris	AIX 5L	Linux
pd_dbsync_lck_release_count	Y	Y	Y	Y
pd_dbsync_point	Y	Y	Y	Y
pd_db_io_error_action	Y	Y	Y	Y
pd_deadlock_priority_use	Y	Y	Y	Y
pd_debug_info_netstat	Y	Y	Y	Y
pd_dec_sign_normalize	Y	Y	Y	Y
pd_def_buf_control_area_assign	Y	Y	Y	Y
pd_delete_reserved_word_file	Y	Y	Y	Y
pd_dfw_awt_process	Y	Y	Y	Y
pd_dfw_syncpoint_skip_limit	Y	Y	Y	Y
pd_dic_shmpool_size	Y	Y	Y	Y
pd_directory_server	C	C	C	N
pd_down_watch_proc	Y	Y	Y	Y
pd_dump_suppress_watch_time	Y	Y	Y	Y
pd_fes_lck_pool_size	Y	Y	Y	Y
pd_floatable_bes	Y	Y	Y	Y
pd_foreign_server_libpath	C	N	C	N
pd_ha	Y	Y	Y	Y
pd_ha_acttype	Y	Y	Y	Y
pd_ha_agent	Y	Y	Y	Y
pd_ha_ipaddr_inherit	Y	Y	Y	Y
pd_ha_max_act_guest_servers	Y	Y	Y	Y
pd_ha_max_server_process	Y	Y	Y	Y
pd_ha_mgr_rerun	Y	Y	Y	Y
pd_ha_prc_cleanup_check	Y	N	Y	Y
pd_ha_process_count	Y	Y	Y	Y
pd_ha_resource_act_wait_time	Y	Y	Y	Y

Operand	HP-UX	Solaris	AIX 5L	Linux
pd_ha_restart_failure	Y	Y	Y	Y
pd_ha_server_process_standby	Y	Y	Y	Y
pd_ha_switch_timeout	Y	Y	Y	Y
pd_ha_transaction	Y	Y	Y	Y
pd_ha_trn_queuing_wait_time	Y	Y	Y	Y
pd_ha_trn_restart_retry_time	Y	Y	Y	Y
pd_ha_unit	Y	Y	Y	Y
pd_hash_table_size	Y	Y	Y	Y
pd_hashjoin_hashing_mode	Y	Y	Y	Y
pd_hostname	Y	Y	Y	Y
pd_host_watch_interval	Y	Y	Y	Y
pd_indexlock_mode	Y	Y	Y	Y
pd_inner_replica_control	Y	Y	Y	Y
pd_ipc_conn_count	Y	Y	Y	Y
pd_ipc_conn_interval	Y	Y	Y	Y
pd_ipc_conn_nblock	Y	Y	Y	Y
pd_ipc_conn_nblock_time	Y	Y	Y	Y
pd_ipc_inet_bufsize	Y	Y	Y	Y
pd_ipc_recv_count	Y	Y	Y	Y
pd_ipc_send_count	Y	Y	Y	Y
pd_ipc_send_retrycount	Y	Y	Y	Y
pd_ipc_send_retrysleeptime	Y	Y	Y	Y
pd_ipc_tcp_nodelayack	N	N	Y	N
pd_ipc_unix_bufsize	Y	Y	Y	Y
pd_java_archive_directory	C	C	C	Y
pd_java_classpath	C	C	C	Y
pd_java_libpath	C	C	C	Y

Operand	HP-UX	Solaris	AIX 5L	Linux
pd_java_option	C	C	C	Y
pd_java_routine_stack_size	C	C	C	Y
pd_java_runtimepath	C	C	C	Y
pd_java_stdout_file	C	C	C	Y
pd_jp1_event_level	Y	C	C	Y
pd_jp1_event_msg_out	Y	C	C	Y
pd_jp1_use	Y	C	C	Y
pd_key_resource_type	Y	Y	Y	Y
pd_large_file_use	Y	Y	Y	Y
pd_lck_deadlock_info	Y	Y	Y	Y
pd_lck_hash_entry	Y	Y	Y	Y
pd_lck_pool_size	Y	Y	Y	Y
pd_lck_queue_limit	Y	Y	Y	Y
pd_lck_release_detect	Y	Y	Y	Y
pd_lck_release_detect_interval	Y	Y	Y	Y
pd_lck_until_disconnect_cnt	Y	Y	Y	Y
pd_lck_wait_timeout	Y	Y	Y	Y
pd_list_initialize_timing	Y	Y	Y	Y
pd_log_auto_unload_path	Y	Y	Y	Y
pd_log_dual	Y	Y	Y	Y
pd_log_dual_write_method	N	N	Y	N
pd_log_max_data_size	Y	Y	Y	Y
pd_log_org_no_standby_file_opr	Y	Y	Y	Y
pd_log_org_reflected_logpoint	Y	Y	Y	Y
pd_log_rec_leng	Y	Y	Y	Y
pd_log_remain_space_check	Y	Y	Y	Y
pd_log_rerun_reserved_file_open	Y	Y	Y	Y

Operand	HP-UX	Solaris	AIX 5L	Linux
pd_log_rerun_swap	Y	Y	Y	Y
pd_log_rpl_no_standby_file_opr	Y	Y	Y	Y
pd_log_sdinterval	Y	Y	Y	Y
pd_log_singleoperation	Y	Y	Y	Y
pd_log_swap_timeout	Y	Y	Y	Y
pd_log_unload_check	Y	Y	Y	Y
pd_log_write_buff_count	Y	Y	Y	Y
pd_lv_mirror_use	Y	Y	Y	Y
pd_master_file_name	Y	Y	Y	Y
pd_max_access_tables	Y	Y	Y	Y
pd_max_access_tables_wrn_pnt	Y	Y	Y	Y
pd_max_add_dbbuff_no	Y	Y	Y	Y
pd_max_add_dbbuff_shm_no	Y	Y	Y	Y
pd_max_ard_process	Y	Y	Y	Y
pd_max_bes_process	Y	Y	Y	Y
pd_max_commit_write_reclaim_no	Y	Y	Y	Y
pd_max_dic_process	Y	Y	Y	Y
pd_max_file_no	Y	Y	Y	Y
pd_max_file_no_wrn_pnt	Y	Y	Y	Y
pd_max_foreign_server	C	C	C	C
pd_max_list_users	Y	Y	Y	Y
pd_max_list_count	Y	Y	Y	Y
pd_max_list_users_wrn_pnt	Y	Y	Y	Y
pd_max_list_count_wrn_pnt	Y	Y	Y	Y
pd_max_open_fds	Y	Y	Y	Y
pd_max_open_holdable_cursors	Y	Y	Y	Y
pd_max_rdarea_no	Y	Y	Y	Y

Operand	HP-UX	Solaris	AIX 5L	Linux
pd_max_rdarea_no_wrn_pnt	Y	Y	Y	Y
pd_max_recover_process	Y	Y	Y	Y
pd_max_reflect_process_count	Y	Y	Y	Y
pd_max_server_process	Y	Y	Y	Y
pd_max_users	Y	Y	Y	Y
pd_max_users_wrn_pnt	Y	Y	Y	Y
pd_mlg_file_size	Y	Y	Y	Y
pd_mlg_msg_log_unit	Y	Y	Y	Y
pd_mode_conf	Y	Y	Y	Y
pd_module_trace_max	Y	Y	Y	Y
pd_module_trace_timer_level	Y	Y	Y	Y
pd_name_port	Y	Y	Y	Y
pd_node_name	Y	N	Y	N
pd_non_floatable_bes	Y	Y	Y	Y
pd_nowait_scan_option	Y	Y	Y	Y
pd_oltp_holdcr	Y	Y	Y	Y
pd_optimize_level	Y	Y	Y	Y
pd_overflow_suppress	Y	Y	Y	Y
pd_pageaccess_mode	Y	Y	Y	Y
pd_plugin_ixmk_dir	Y	Y	Y	Y
pd_process_count	Y	Y	Y	Y
pd_process_terminator	Y	Y	Y	Y
pd_process_terminator_max	Y	Y	Y	Y
pd_queue_watch_time	Y	Y	Y	Y
pd_queue_watch_timeover_action	Y	Y	Y	Y
pd_rdarea_list_no_wrn_pnt	Y	Y	Y	Y
pd_rdarea_open_attribute	Y	Y	Y	Y

Operand	HP-UX	Solaris	AIX 5L	Linux
pd_rdarea_open_attribute_use	Y	Y	Y	Y
pd_rdarea_warning_point	Y	Y	Y	Y
pd_redo_allpage_put	Y	Y	Y	Y
pd_reduced_check_time	Y	Y	Y	Y
pd_registered_port	N	N	N	Y
pd_registered_port_check	N	N	N	Y
pd_registered_port_level	N	N	N	Y
pd_registry_cache_size	Y	Y	Y	Y
pd_rise_disaster_mode	Y	N	Y	N
pd_rise_fence_level	Y	N	Y	N
pd_rise_pairvolume_combination	Y	N	Y	N
pd_rise_use	Y	N	Y	N
pd_rorg_predict	Y	Y	Y	Y
pd_routine_def_cache_size	Y	Y	Y	Y
pd_rpc_trace	Y	Y	Y	Y
pd_rpc_trace_name	Y	Y	Y	Y
pd_rpc_trace_size	Y	Y	Y	Y
pd_rpl_hdepath	Y	Y	Y	Y
pd_rpl_init_start	Y	Y	Y	Y
pd_rpl_reflect_mode	Y	Y	Y	Y
pd_sds_shmpool_size	Y	Y	Y	Y
pd_server_cleanup_interval	Y	Y	Y	Y
pd_server_entry_queue	Y	Y	Y	Y
pd_service_port	Y	Y	Y	Y
pd_shared_rdarea_use	Y	Y	Y	Y
pd_shmpool_attribute	Y	Y	C	Y
pd_space_level	Y	Y	Y	Y

Operand	HP-UX	Solaris	AIX 5L	Linux
pd_spd_assurance_count	Y	Y	Y	Y
pd_spd_assurance_msg	Y	Y	Y	Y
pd_spd_dual	Y	Y	Y	Y
pd_spd_max_data_size	Y	Y	Y	Y
pd_spd_reduced_mode	Y	Y	Y	Y
pd_spd_reserved_file_auto_open	Y	Y	Y	Y
pd_spd_syncpoint_skip_limit	Y	Y	Y	Y
pd_spool_cleanup	Y	Y	Y	Y
pd_spool_cleanup_interval	Y	Y	Y	Y
pd_spool_cleanup_interval_level	Y	Y	Y	Y
pd_spool_cleanup_level	Y	Y	Y	Y
pd_sql_object_cache_size	Y	Y	Y	Y
pd_sql_send_buff_size	Y	Y	Y	Y
pd_start_level	Y	Y	Y	Y
pd_start_skip_unit	Y	Y	Y	Y
pd_start_time_out	Y	Y	Y	Y
pd_statistics	Y	Y	Y	Y
pd_stj_file_size	Y	Y	Y	Y
pd_stj_buff_size	Y	Y	Y	Y
pd_sts_file_name_1-7	Y	Y	Y	Y
pd_sts_initial_error	Y	Y	Y	Y
pd_sts_last_active_file	Y	Y	Y	Y
pd_sts_last_active_side	Y	Y	Y	Y
pd_sts_last_active_side_sub	C	N	Y	N
pd_sts_last_active_subfile	C	N	Y	N
pd_sts_singleoperation	Y	Y	Y	Y
pd_sts_subfile_name_1 to 7	C	N	Y	N

Operand	HP-UX	Solaris	AIX 5L	Linux
pd_substr_length	Y	Y	Y	Y
pd_svr_castoff_size	Y	Y	Y	N
pd_sysdef_default_option	Y	Y	Y	Y
pd_syssts_file_name_1-7	Y	Y	Y	Y
pd_syssts_initial_error	Y	Y	Y	Y
pd_syssts_last_active_file	Y	Y	Y	Y
pd_syssts_last_active_side	Y	Y	Y	Y
pd_syssts_last_active_side_sub	C	N	Y	N
pd_syssts_last_active_side_sub	C	N	Y	N
pd_syssts_singleoperation	Y	Y	Y	Y
pd_syssts_subfile_name_1 to 7	C	N	Y	N
pd_system_complete_wait_time	Y	Y	Y	Y
pd_system_dbsync_point	Y	Y	Y	Y
pd_system_id	Y	Y	Y	Y
pd_table_def_cache_size	Y	Y	Y	Y
pd_tcp_inet_bufsize	Y	Y	Y	Y
pd_tcp_unix_bufsize	Y	Y	Y	Y
pd_term_watch_count	Y	Y	Y	Y
pd_thdlock_pipe_retry_interval	Y	Y	Y	Y
pd_thdlock_retry_time	Y	Y	Y	Y
pd_thdlock_sleep_func	Y	Y	Y	Y
pd_thdlock_wakeup_lock	Y	Y	Y	Y
pd_thdspnlk_spn_count	Y	Y	Y	Y
pd_thred_max_stack_size	Y	Y	Y	Y
pd_trn_commit_optimize	Y	Y	Y	Y
pd_trn_rerun_branch_auto_decide	Y	Y	Y	Y
pd_trn_send_decision_interval	Y	Y	Y	Y

Operand	HP-UX	Solaris	AIX 5L	Linux
pd_trn_send_decision_intval_sec	Y	Y	Y	Y
pd_trn_send_decision_retry_time	Y	Y	Y	Y
pd_trn_watch_time	Y	Y	Y	Y
pd_type_def_cache_size	Y	Y	Y	Y
pd_uap_exerror_log_dir	Y	Y	Y	Y
pd_uap_exerror_log_param_size	Y	Y	Y	Y
pd_uap_exerror_log_size	Y	Y	Y	Y
pd_uap_exerror_log_use	Y	Y	Y	Y
pd_uap_wait	Y	Y	Y	Y
pd_unit_id	Y	Y	Y	Y
pd_utl_buff_size	Y	Y	Y	Y
pd_utl_exec_mode	Y	Y	Y	Y
pd_utl_exec_time	Y	Y	Y	Y
pd_view_def_cache_size	Y	Y	Y	Y
pd_watch_pc_client_time	Y	Y	Y	Y
pd_watch_resource	Y	Y	Y	Y
pd_watch_time	Y	Y	Y	Y
pd_work_buff_expand_limit	Y	Y	Y	Y
pd_work_buff_mode	Y	Y	Y	Y
pd_work_buff_size	Y	Y	Y	Y
pd_work_table_option	Y	Y	Y	Y
pdwork_wrn_pnt	Y	Y	Y	Y
pdbuffer	Y	Y	Y	Y
pdcltgrp	Y	Y	Y	Y
PDCONFPATH	Y	Y	Y	Y
pdhagroup	Y	Y	Y	Y
pdhibegin	Y	Y	Y	Y

Operand	HP-UX	Solaris	AIX 5L	Linux
pdhubopt	C	C	C	C
pdlbuffer	Y	Y	Y	Y
pdlogadfg -d spd	Y	Y	Y	Y
pdlogadfg -d ssp	C	N	Y	N
pdlogadfg -d sys	Y	Y	Y	Y
pdlogadpf -d spd	Y	Y	Y	Y
pdlogadpf -d ssp	C	N	Y	N
pdlogadpf -d sys	Y	Y	Y	Y
pdmlgput	Y	Y	Y	Y
pdplgprm	Y	Y	Y	Y
pdplugin	Y	Y	Y	Y
pdstart	Y	Y	Y	Y
pdstbegin	Y	Y	Y	Y
pdunit	Y	Y	Y	Y
pdwork	Y	Y	Y	Y
SHMMAX	Y	Y	Y	Y
TZ	Y	Y	Y	Y

Y: Operand is specifiable.

N: Operand is not specifiable.

C: Whether or not the operand can be specified depends on the condition. For details, see the description of each operand.

I. List of Operands That Can Be Specified When Using the Standby-less System Switchover (Effects Distributed) Facility (Unit Control Information Definition)

When using the standby-less system switchover (effects distributed) facility, you can specify the following operands in the unit control information definition. You cannot specify any other operands in the unit control information definition. If any other operand is specified, HiRDB cannot start (the message `KFPS05618-E` is output).

- `pd_down_watch_proc`
- `pd_ha_acttype`
- `pd_ha_agent`
- `pd_ha_max_act_guest_servers`
- `pd_ha_max_server_process`
- `pd_ha_process_count`
- `pd_ha_resource_act_wait_time`
- `pd_ha_unit`
- `pd_hostname`
- `pd_rpl_hdepath`
- `pd_syssts_file_name_1` to `7`
- `pd_syssts_initial_error`
- `pd_syssts_last_active_file`
- `pd_syssts_last_active_side`
- `pd_syssts_singleoperation`
- `pd_unit_id`

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)