FUJITSU

FUJITSU Enterprise Postgres 9.4

Operation Guide

Linux

J2UL-2037-01ENZ0(00)
July 2015

Preface

Purpose of this document

The Enterprise Postgres database system extends the PostgreSQL features and runs on the Linux platform.

This document is the Enterprise Postgres Operation Guide.

Intended readers

This document is intended for those who install and operate Enterprise Postgres.

Readers of this document are assumed to have general knowledge of:

- PostgreSQL
- SQL
- Linux

Structure of this document

This document is structured as follows:

Appendix F Notes on PL/extJava

Describes the note on PL/extJava.

Appendix G PL/extJava Log Information

Describes the PL/extJava log information.

Export restrictions

Exportation/release of this document may require necessary procedures in accordance with the regulations of your resident country and/or US export control laws.

Issue date and version

```
First edition: July 2015
```

Copyright

Contents

Chapter 1 Operating Enterprise Postgres

This chapter describes how to operate Enterprise Postgres.

1.1 Operating Methods

There are two methods of managing Enterprise Postgres operations:

- Operation management using GUI tools
- Operation management using commands

See

Before performing switchover or failover operation using database multiplexing, refer to "Database Multiplexing Mode" in the Cluster Operation Guide.

Operation management using GUI tools

This involves managing operations using the WebAdmin and pgAdmin GUI tools.

- Management using WebAdmin

  This removes the requirement for complex environment settings and operational design for backup and recovery that is usually required for running a database. It enables you to easily and reliably monitor the state of the database, back up the database, and restore it even if you do not have expert knowledge of databases.

- Management using pgAdmin

  When developing applications and maintaining the database, you can use pgAdmin to perform simple operations on database objects, such as:

  - Rebuild indexes and update statistics
  - Create, delete, and update database objects

  In addition, from pgAdmin of Enterprise Postgres, you can use the expanded features provided by Enterprise Postgres on the PostgreSQL SQL commands.

  See

  Refer to pgAdmin Help for information on the expanded features of pgAdmin provided by Enterprise Postgres.

Operation management using commands

You can use commands for configuring and operating the database and managing operations. However, note that if you start managing operations using commands, you cannot switch to WebAdmin-based operation management.

Note

You cannot combine WebAdmin and server commands to perform the following operations:

- Use WebAdmin to operate an instance created using the initdb command
- Use commands to operate an instance created using WebAdmin
- Use WebAdmin to recover a database backed up using commands

For instances created with WebAdmin, however, backup can be obtained with the pgx_dmpall command. Also, WebAdmin can perform recovery by using the backup obtained with the pgx_dmpall command.

- You can perform backup and restoration in pgAdmin, but the backup data obtained with WebAdmin and pgx_dumpall is not compatible with the backup data obtained with pgAdmin.
- Refer to pgAdmin Help for other notes on pgAdmin.

Features used in each phase

The following table lists the features used in each phase for GUI-based operations and command-based operations.

Operation		GUI-based operation	Command-based operation
Setup	Instance creation	WebAdmin	initdb command
	Modification of the configuration file	WebAdmin	Directly edit the configuration file
Instance start		WebAdmin	pg_ctl command
Database creation		pgAdmin	Specify using the DDL statement, and define using psql and applications
Database backup		WebAdmin pgx_dmpall command	pgx_dmpall command
Monitoring	Database failure	WebAdmin(*1)	Messages output to the system log (*1)
	Disk space	WebAdmin (*1) (*2)	OS-provided df command (*1)
	Connection status	pgAdmin	psql command (*3)
Database recovery		WebAdmin	pgx_rcvall command

*1: Operations can be monitored using operation management middleware (such as Systemwalker Centric Manager).

*2: A warning is displayed when disk usage reaches 80%.

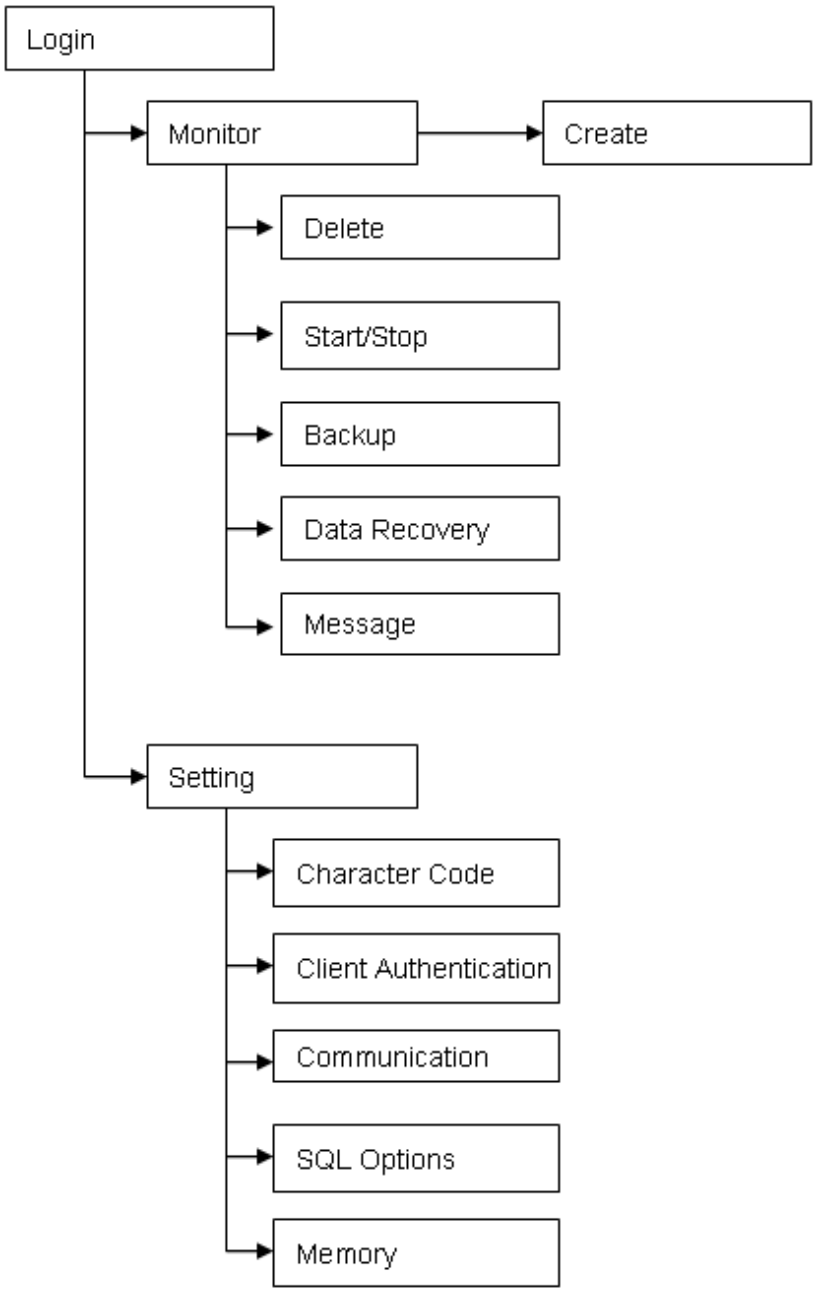
*3: This command searches for pg_stat_activity in the standard statistics views and monitors the state.

1.2 Activating WebAdmin

This section describes how to activate and log in to WebAdmin.

1.2.1 Flow of WebAdmin

The figure below shows the flow of WebAdmin GUI windows.



Monitor menu

Using this menu, you can operate the following instances and display their states:

- [Create]: Creates a database cluster and instance
- [Delete]: Deletes a database cluster and instance
- [Start/Stop]: Starts or stops an instance
- [Backup]: Performs back up of a database cluster
- [Data Recovery]: Recovers a database cluster
- [Message]: Displays messages about operations performed using WebAdmin, and about errors that are detected

See

Refer to the following for information on the functionality available from the [Monitor] menu:

- Creation or deletion: "Creating an Instance" in the Installation and Setup Guide for Server
- Starting and stopping: "2.1.1 Using WebAdmin"
- Backup: "3.2.1 Using WebAdmin"
- Data recovery: "8.3.1 Using WebAdmin"

Setting menu

Using this menu, you can set the definition information for the following instances:

- [Character Code]: Sets the character set and locale
- [Client Authentication]: Sets the authentication information to be used when a client connects to an instance
- [Communication]: Sets the communication definition for applications and instances
- [SQL Options]: Sets the definition to be used when executing an SQL statement
- [Memory]: Sets the memory to be used

See

Refer to "Changing the settings" in the Installation and Setup Guide for Server for information on the [Setting] menu.

1.2.2 Logging in to WebAdmin

This section describes how to log in to WebAdmin.

User environment

The following browser is required for using WebAdmin:

- Internet Explorer 8.0 or later

Activation URL for WebAdmin

In the browser address bar, type the activation URL of the WebAdmin window in the following format:

```
http://hostNameOrIpAddress:portNumber/
```

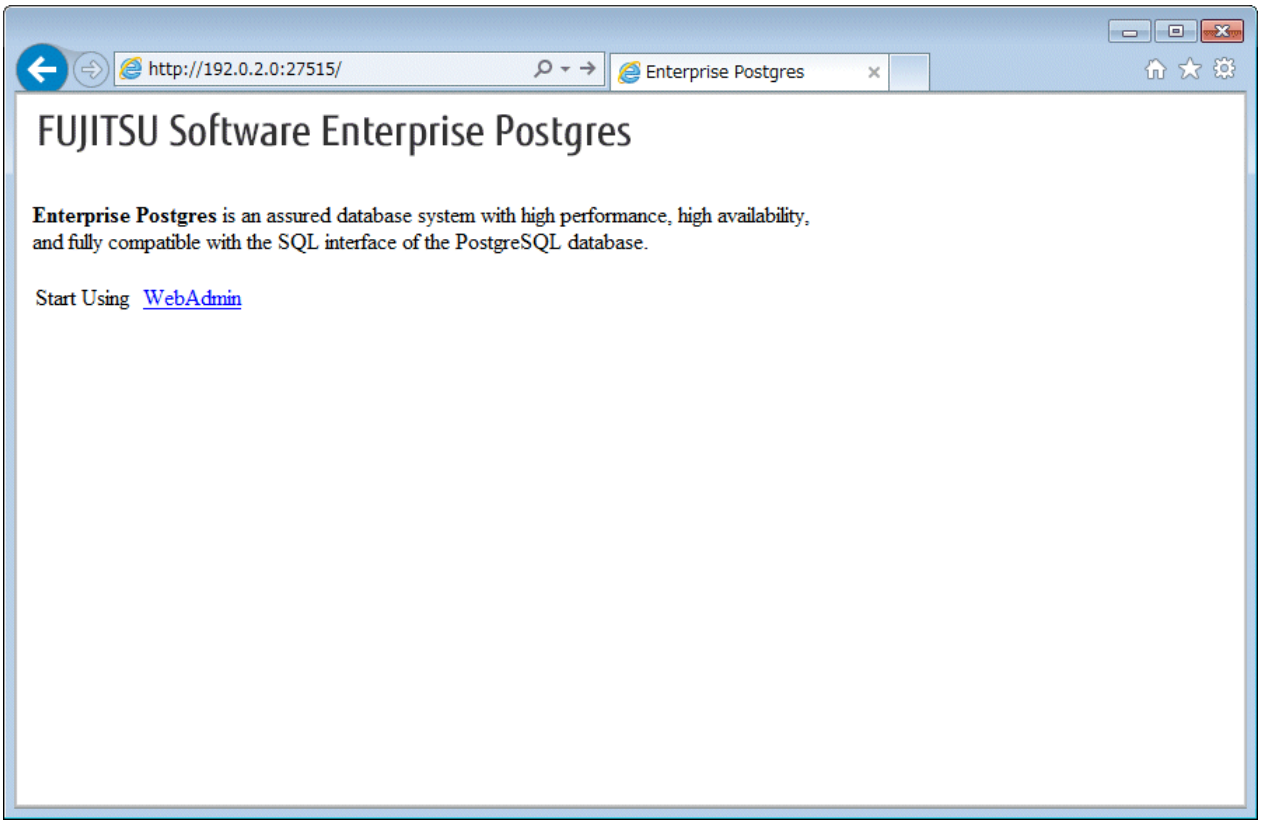
- *hostNameOrIpAddress*: The host name or IP address of the server where Enterprise Postgres is installed.
- *portNumber*: The port number of WebAdmin. The default port number is 27515.

Example

For a server with IP address "192.0.2.0" and port number "27515"

```
http://192.0.2.0:27515/
```

The activation URL window shown below is displayed.



Point

- You must activate the Web server feature of WebAdmin before using WebAdmin.
- Refer to "Appendix D Activating and Stopping the Web Server Feature of WebAdmin" for information on how to activate the Web server feature of WebAdmin.

Log in to WebAdmin

Click [Enterprise Postgres WebAdmin] in the activation URL window to activate WebAdmin and display the [Log in] window. You can log in to WebAdmin using the [Log in] window.

To log in, specify the following values:

- [User ID]: User ID (OS user account) of the instance administrator
- [Password]: Password corresponding to the user ID

Point

Use the OS user account as the user ID of the instance administrator. Refer to "Creating an Instance Administrator" in the Installation and Setup Guide for Server for details.

1.3 Starting pgAdmin

This section describes how to start pgAdmin, how to add an instance required for managing a database, and how to connect to and disconnect from the instance.

You can use pgAdmin on the Windows client.

1.3.1 Starting pgAdmin

This section explains how to start pgAdmin if you are using it from the product "Enterprise Postgres Client (*AA*bit) *x.y.z*" (where *AA* is "32" or "64", *x.y.z* is the version number(*x.y* SP*z*)).

Windows(R) 8 or Windows Server(R) 2012

From the [Start] screen, start [pgAdmin III(*AA*bit)(*x.y.z*)].

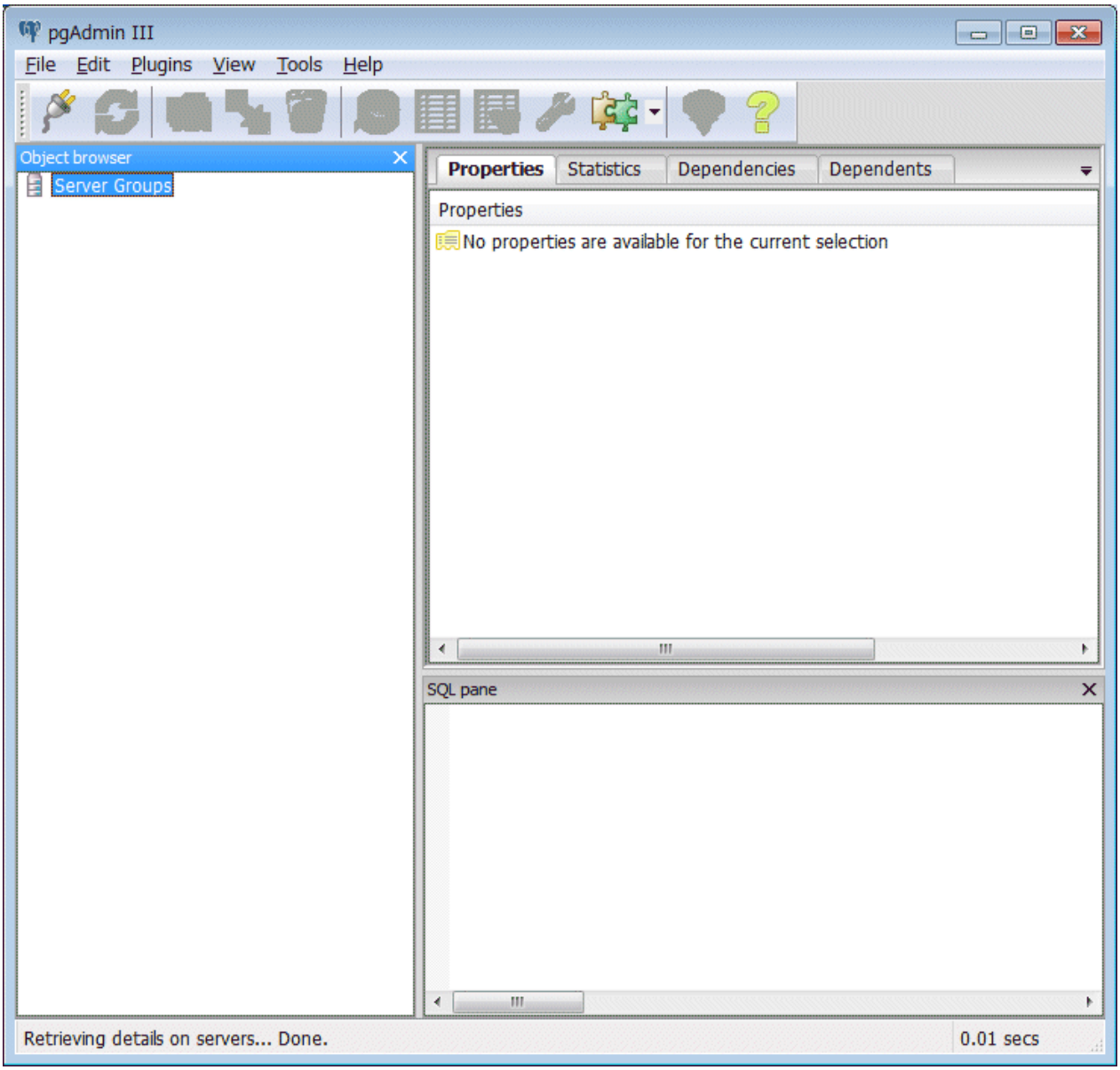
Windows(R) 8.1 or Windows Server(R) 2012 R2

From the [Apps] view, start [pgAdmin III(*AA*bit)(*x.y.z*)].

Other operating systems

Click [Start] >> [All Programs] >> [Enterprise Postgres Client (*AA*bit) *x.y.z*] and start [pgAdmin III(*AA*bit)(*x.y.z*)].

The following window is displayed when pgAdmin starts.



Note

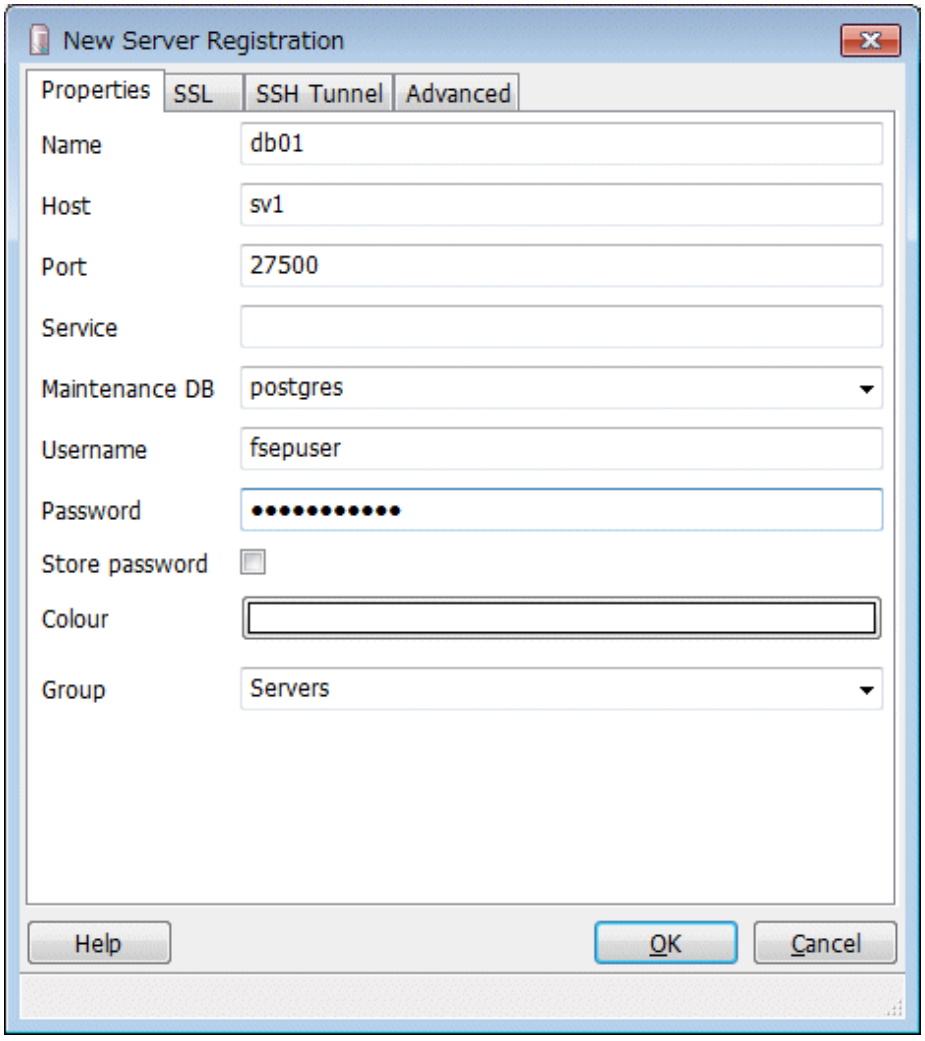
- You must start the instance to be connected to before using pgAdmin.
- Refer to "2.1 Starting and Stopping an Instance" for information on how to start an instance.
- Adobe(R) Reader(R) X is required for browsing the manual from [Enterprise Postgres Help] in pgAdmin.

1.3.2 Adding an Instance

This section describes how to add an instance to be connected to.

1. From the [File] menu in pgAdmin, click [Add Server].

2. In the [New Server Registration] window, specify a value for each item.



([Properties] tab)

- [Name]: Name of the instance to be managed
- [Host]: Host name or IP address of the server where Enterprise Postgres is installed
- [Port]: Port number of the instance
- [Username]: User ID of the instance administrator
- [Password]: Password for the user ID specified in [Username]

When you add an instance using pgAdmin, the instance is automatically connected to immediately after the addition is completed.

Note

If you select [Store password], a file storing the Enterprise Postgres connection password is created in the following location. Set the appropriate access permissions for the password file to protect it from unauthorized access.

- %APPDATA%\postgresql\pgpass.conf

1.3.3 Connecting/Disconnecting an Instance

This section describes how to connect pgAdmin to an instance, and how to disconnect it.

Note

To connect to an instance created using WebAdmin, you must first configure the settings in the [Client Authentication] window of WebAdmin to permit connection from pgAdmin.

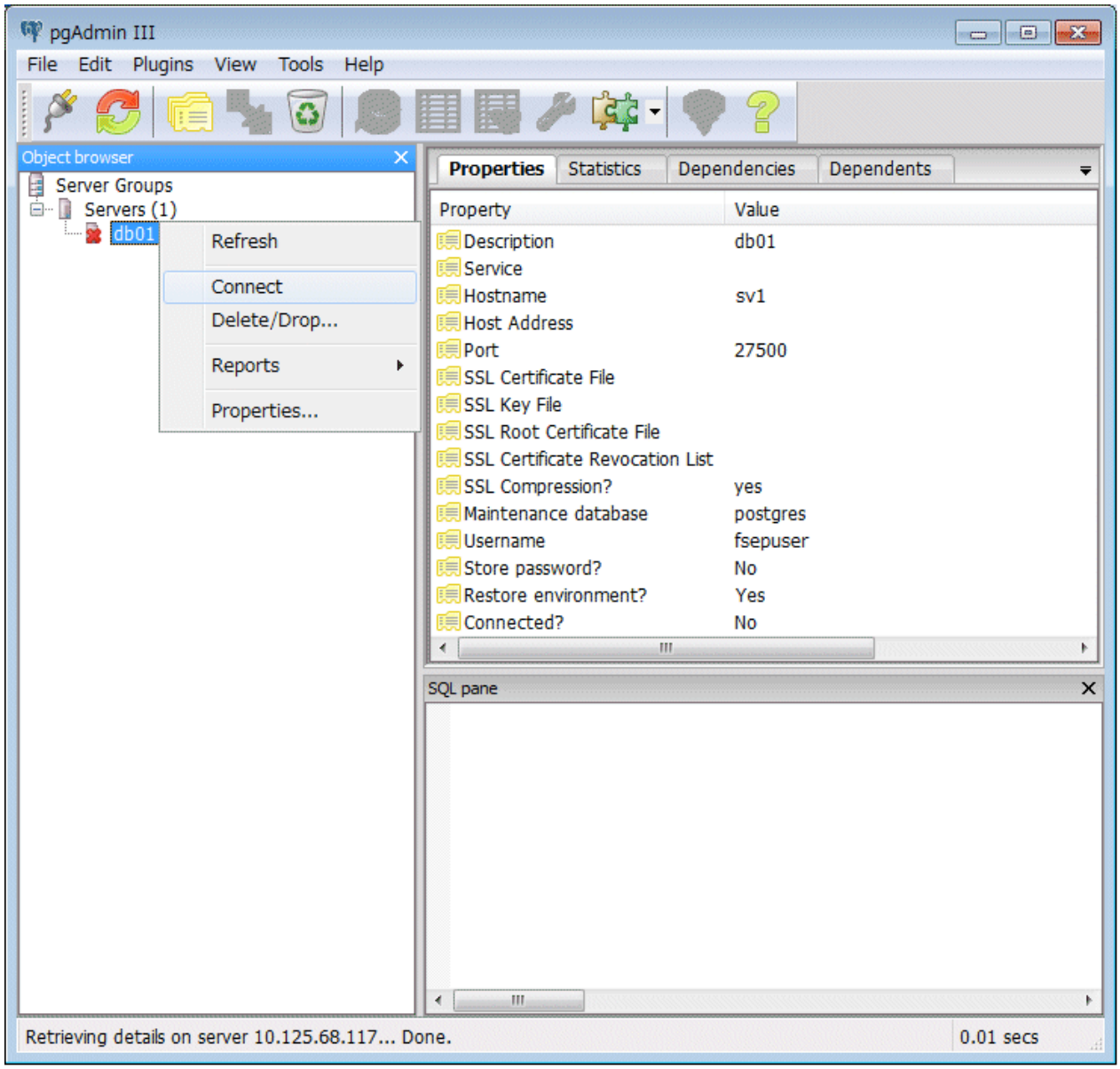
See

Refer to "Changing the settings" in the Installation and Setup Guide for Server for information on the [Client Authentication] window of WebAdmin.

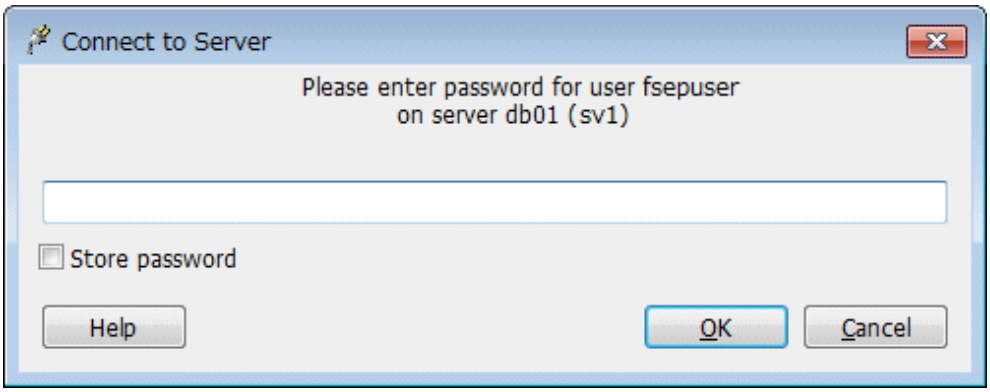
Connecting to an instance

Starting pgAdmin does not connect it to any instance.

To connect to an instance, right-click the instance in [Object browser] and select [Connect].



If a password was not saved when the instance was added, the following password entry window is displayed.



Disconnecting from an instance

To disconnect from an instance, right-click the server in [Object browser] in the pgAdmin window and select [Disconnect server].

1.4 Operations Using Commands

You can operate and manage the database using the following commands:

- Server commands

  This group of commands includes commands for creating a database cluster and controlling the database. You can run these commands on the server where the database is operating.

  To use these commands, you must configure the environment variables.

  See

  - Refer to "PostgreSQL Server Applications" under "Reference" in the PostgreSQL Documentation, or "Reference" for information on server commands.
  - Refer to "Configure the environment variables" under the procedure for creating an instance in "Using the initdb Command" in the Installation and Setup Guide for Server for information on the values to be set in the environment variables.

- Client commands

  This group of commands includes the psql command and commands for extracting the database cluster to a script file. These commands can be executed on the client that can connect to the database, or on the server on which the database is running.

  To use these commands, you need to configure the environment variables.

  See

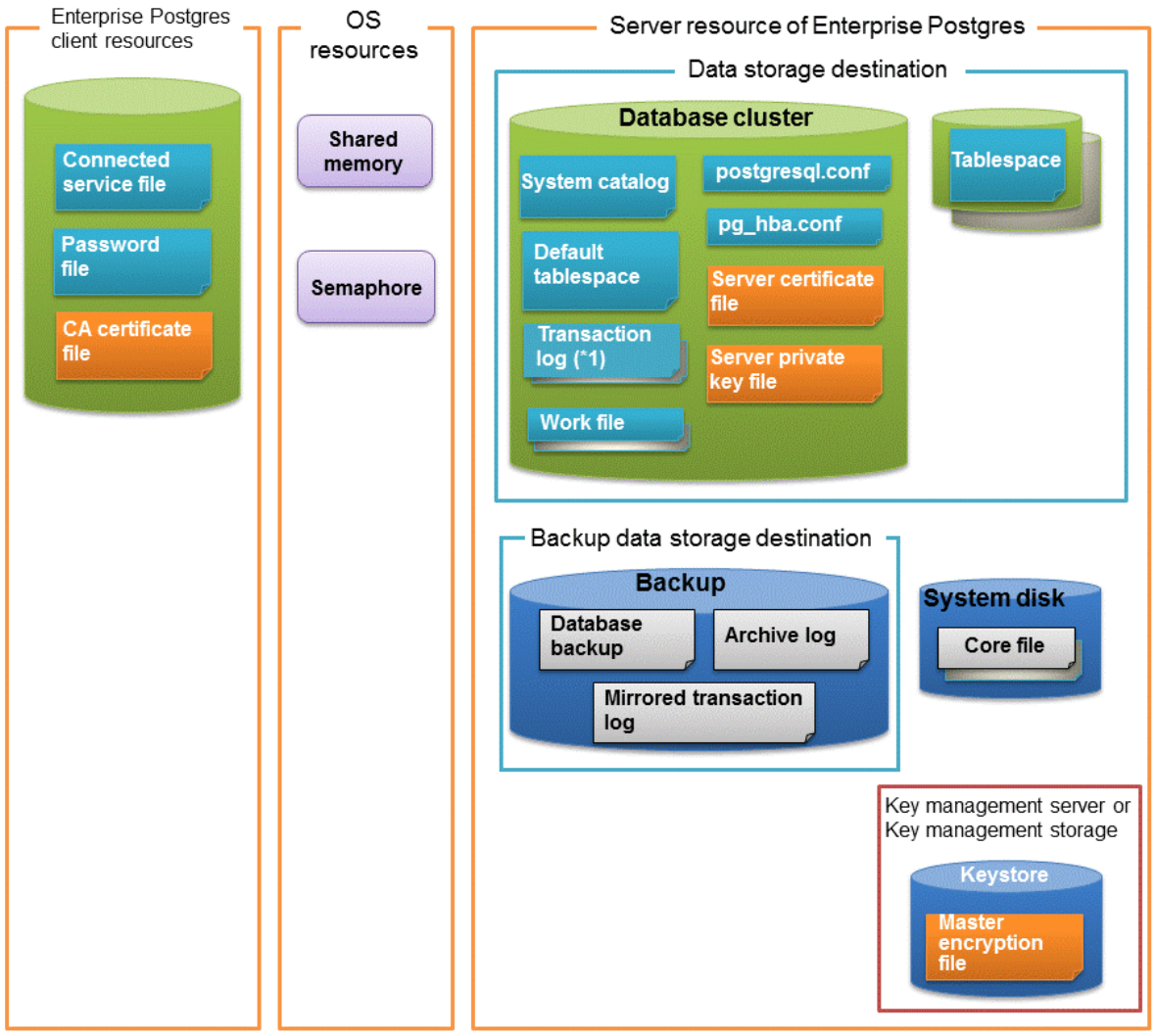
  - Refer to "PostgreSQL Client Applications" under "Reference" in the PostgreSQL Documentation, or "Reference" for information on client commands.
  - Refer to "Configuring Environment Variables" in the Installation and Setup Guide for Client for information on the values to be set in the environment variables.

1.5 Operating Environment of Enterprise Postgres

This section describes the operating environment and the file composition of Enterprise Postgres.

1.5.1 Operating Environment

The following figure shows the configuration of the Enterprise Postgres operating environment. The tables given below list the roles of the OS resources and Enterprise Postgres resources.



*1: To distribute the I/O load, place the transaction log on a different disk from the data storage destination.

Table 1.1 OS resources

Type	Role
Shared memory	Used when a database process exchanges information with an external process.
Semaphore	

Table 1.2 Enterprise Postgres client resources

Type	Role
Connection service file	Specifies information, such as the host name, user ID, and password, for connecting to Enterprise Postgres
Password file	Securely manages the password for connecting to Enterprise Postgres
CA certificate file	CA (certificate authority) certificate used for server authentication when encrypting communication data

Table 1.3 Server resources of Enterprise Postgres

Type	Role
Database cluster	Database storage area on the database storage disk. It is a collection of databases managed by an instance.
System catalog	Contains information required for the system to run, including the database definition information and the operation information created by the user
Default tablespace	Contains table files and index files stored by default
Transaction log	Contains log information in case of a crash recovery or rollback. This is the same as the WAL (Write Ahead Log).
Work file	Work file used when executing applications or commands
postgresql.conf	Contains information that defines the operating environment of Enterprise Postgres
pg_hba.conf	Enterprise Postgres uses this file to authenticate individual client hosts
Server certificate file	Contains information about the server certificate to be used when encrypting communication data and authenticating a server
Server private key file	Contains information about the server private key to be used when encrypting communication data and authenticating a server
Tablespace	Stores table files and index files in a separate area from the database cluster
Backup	Stores the data required for recovering the database when an error, such as disk failure, occurs
Database backup	Contains the backup data for the database
Archive log	Contains the log information for recovery.
Core file	Enterprise Postgres process core file that is output when an error occurs during a Enterprise Postgres process
Key management server or key management storage	Server or storage where the master encryption key file is located
Master encryption key file	Contains the master encryption key to be used when encrypting storage data. The master encryption key file is managed on the key management server or key management storage.

1.5.2 File Composition

Enterprise Postgres consists of the following files for controlling and storing the database. The table below shows the relationship between the number of such files and their location within a single instance.

Table 1.4 Number of files within a single instance and how to specify their location

File type	Required	Quantity	How to specify the location
Program files	Y	Multiple	**64-bit product** /opt/fsepserver64 **32-bit product** /opt/fsepserver32
Database cluster	Y	1	Specify using WebAdmin or server commands.
Tablespace	Y	Multiple	Specify using pgAdmin or the DDL statement.
Backup	Y	Multiple	Specify using WebAdmin or server commands.

File type	Required	Quantity	How to specify the location
Core file	Y	Multiple	Specify using WebAdmin, server commands, or postgresql.conf.
Server certificate file (*1)	N	1	Specify using postgresql.conf.
Server private key file (*1)	N	1	Specify using postgresql.conf.
Master encryption key file (*1)	N	1	Specify the directory created as the key store using postgresql.conf.
Connection service file (*1)	N	1	Specify using environment variables.
Password file (*1)	N	1	Specify using environment variables.
CA certificate file (*1)	N	1	Specify using environment variables.

Y: Mandatory
N: Optional

*1: Set manually when using the applicable feature.

Note

- Do not use an NFS for UNIX-type files used in Enterprise Postgres except when creating a database space in a storage device on a network.
- When PRIMECLUSTER GDS is used, the PRIMECLUSTER GDS disk class cannot deploy the Enterprise Postgres resources below to the root class.
  Deploy these resources to the local class or the shared class.
  - Database cluster
  - Tablespace
  - Backup directory
- If anti-virus software is used, set scan exception settings for directories so that none of the files that comprise Enterprise Postgres are scanned for viruses. Alternatively, if the files that comprise Enterprise Postgres are to be scanned for viruses, stop Enterprise Postgres and perform the scan when tasks that use Enterprise Postgres are not operating.

1.6 Notes on Compatibility of Applications Used for Operations

When you upgrade Enterprise Postgres to a newer version, there may be some affect on applications due to improvements or enhancements in functionality.

Take this into account when creating applications so that you can maintain compatibility after upgrading to a newer version of Enterprise Postgres.

See

Refer to " Notes on Application Compatibility " in the Application Development Guide for details.

Chapter 2 Starting an Instance and Creating a Database

This chapter describes basic operations, from starting an instance to creating a database.

2.1 Starting and Stopping an Instance

This section describes how to start and stop an instance.

- 2.1.1 Using WebAdmin
- 2.1.2 Using Server Commands

Point

To automatically start or stop an instance when the operating system on the database server is started or stopped, refer to "Configuring Automatic Start and Stop of an Instance" in the Installation and Setup Guide for Server and configure the settings.

Note

The collected statistics are initialized if an instance is stopped in the "Immediate" mode or if it is abnormally terminated. To prepare for such initialization of statistics, consider regular collection of the statistics by using the SELECT statement. Refer to "The Statistics Collector" in "Server Administration" in the PostgreSQL Documentation for information on the statistics.

2.1.1 Using WebAdmin

WebAdmin enables you to start or stop an instance and check its operating status.

Starting an instance

Start an instance by using the [Monitor] window in WebAdmin.

The [Start] button is displayed when an instance is stopped.

To start a stopped instance, click [Start].

Stopping an instance

Stop an instance by using the [Monitor] window of WebAdmin.

The [Stop] button is displayed when an instance is active.

To stop an active instance, click [Stop].

Stop mode

Select the mode in which to stop the instance. The following describes the operations of the modes:

Stop mode	Connected clients	Backup being executed using the command
Smart mode (*1)	Waits for all connected clients to be disconnected.	Waits for backups being executed using the command to finish.
Fast mode	Rolls back all transactions being executed and forcibly disconnects clients.	Terminates backups being executed using the command.
Immediate mode	All server processes are terminated immediately. Crash recovery is executed the next time the instance is started.	

*1: When the processing to stop the instance in the Smart mode has started and you want to stop immediately, use the following procedure:

1. Restart the Web server feature of WebAdmin.
2. Log in to WebAdmin again.
3. Click the [Stop] button in the [Monitor] window, and select the Immediate mode to stop the instance.

Checking the operating status of an instance

You can check the operating status of an instance by using the [Monitor] window.

When an instance is started, "Started" is displayed as the operating status. When an instance is stopped, "Stopped" is displayed as the operating status. If an error is detected, an error message is displayed in the message list.

If an instance stops, remove the cause of stoppage and start the instance by using WebAdmin.

Figure 2.1 Status when an instance is active

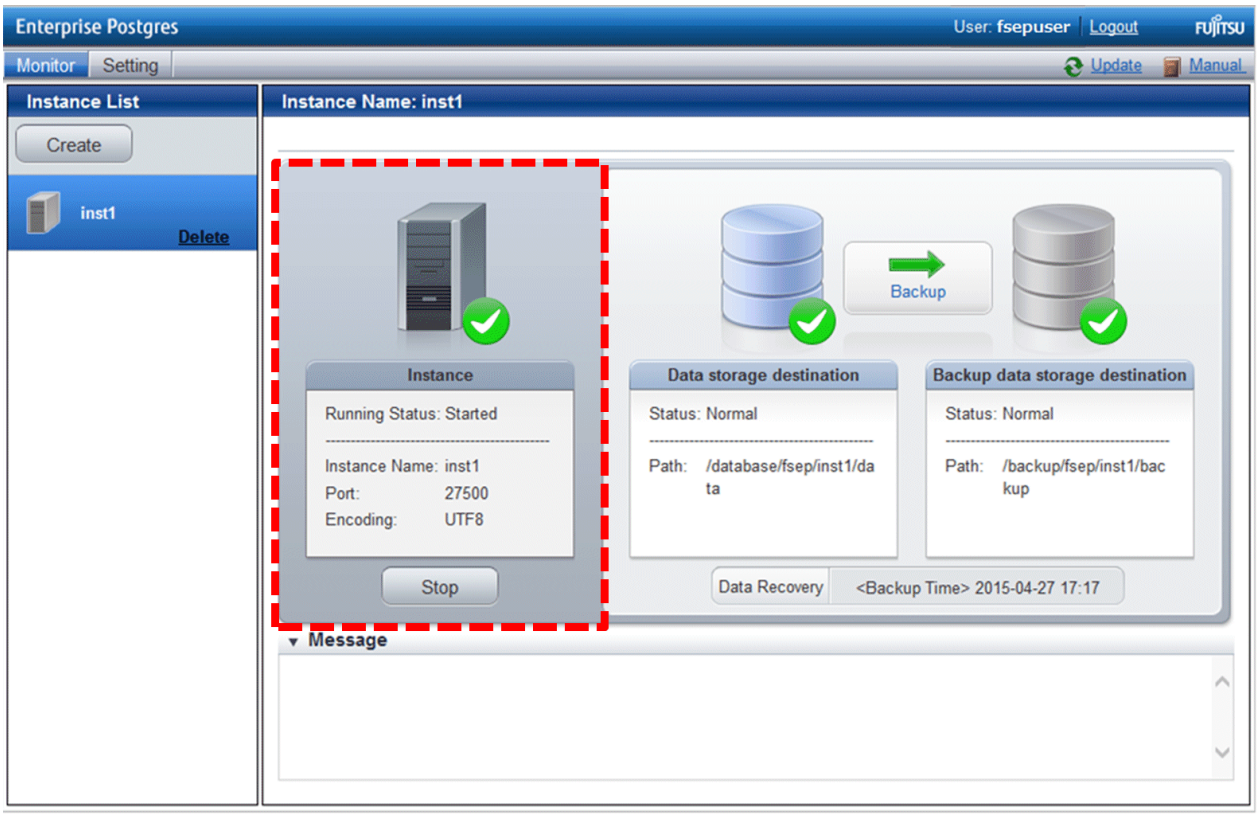
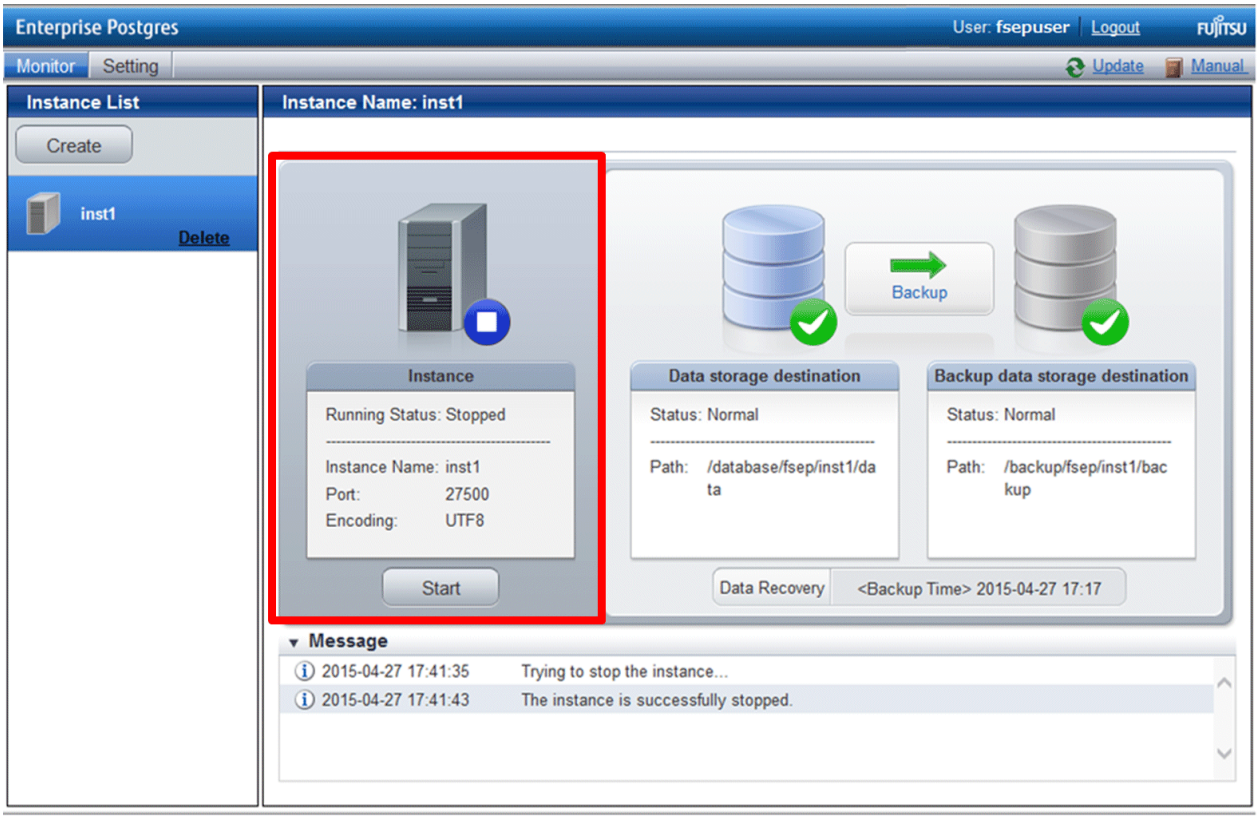


Figure 2.2 Status when an instance is stopped



Note

If an error occurs while communicating with the server, there may be no response from WebAdmin. When this happens, close the browser and then log in again. If this does not resolve the issue, check the system log of the server and confirm whether a communication error has occurred.

2.1.2 Using Server Commands

Server commands enable you to start or stop an instance and check its operating status.

To use sever commands, configure the environment variables.

See

Refer to " Configure the environment variables" in the procedure to create instances in " Using the initdb Command" in the Installation and Setup Guide for Server for information on configuring the environment variables.

Starting an instance

Use the pg_ctl command to start an instance.

Specify the following values in the pg_ctl command:

- Specify "start" as the mode.
- Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

- It is recommended to specify the -w option, which causes the command to return after waiting for the instance to start. If the -w option is not specified, it may not be possible to determine if the starting of the instance completed successfully or if it failed.

  If an application, command, or process tries to connect to the database while the instance is starting up, the message "FATAL:***the database system is starting up(***11189)" is output. However, this message may also be output if the instance is started with the -w option specified.
  This message is output by the pg_ctl command to check if the instance has started successfully. Therefore, ignore this message if there are no other applications, commands, or processes that connect to the database.

Example

```
> pg_ctl start -w -D /database/inst1
```

Stopping an instance

Use the pg_ctl command to stop an instance.

Specify the following values in the pg_ctl command:

- Specify "stop" as the mode.
- Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

Example

```
> pg_ctl stop -D /database/inst1
```

Checking the operating status of an instance

Use the pg_ctl command to check the operating status of an instance.

Specify the following values in the pg_ctl command:

- Specify "status" as the mode.
- Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

Example

When the instance is active:

```
> pg_ctl status -D /database/inst1
pg_ctl: server is running (PID: 1234)
```

When the instance is inactive:

```
> pg_ctl status -D /database/inst1
pg_ctl: no server running.
```

See

Refer to "pg_ctl" under "Reference" in the PostgreSQL Documentation for information on pg_ctl command.

2.2 Creating a Database

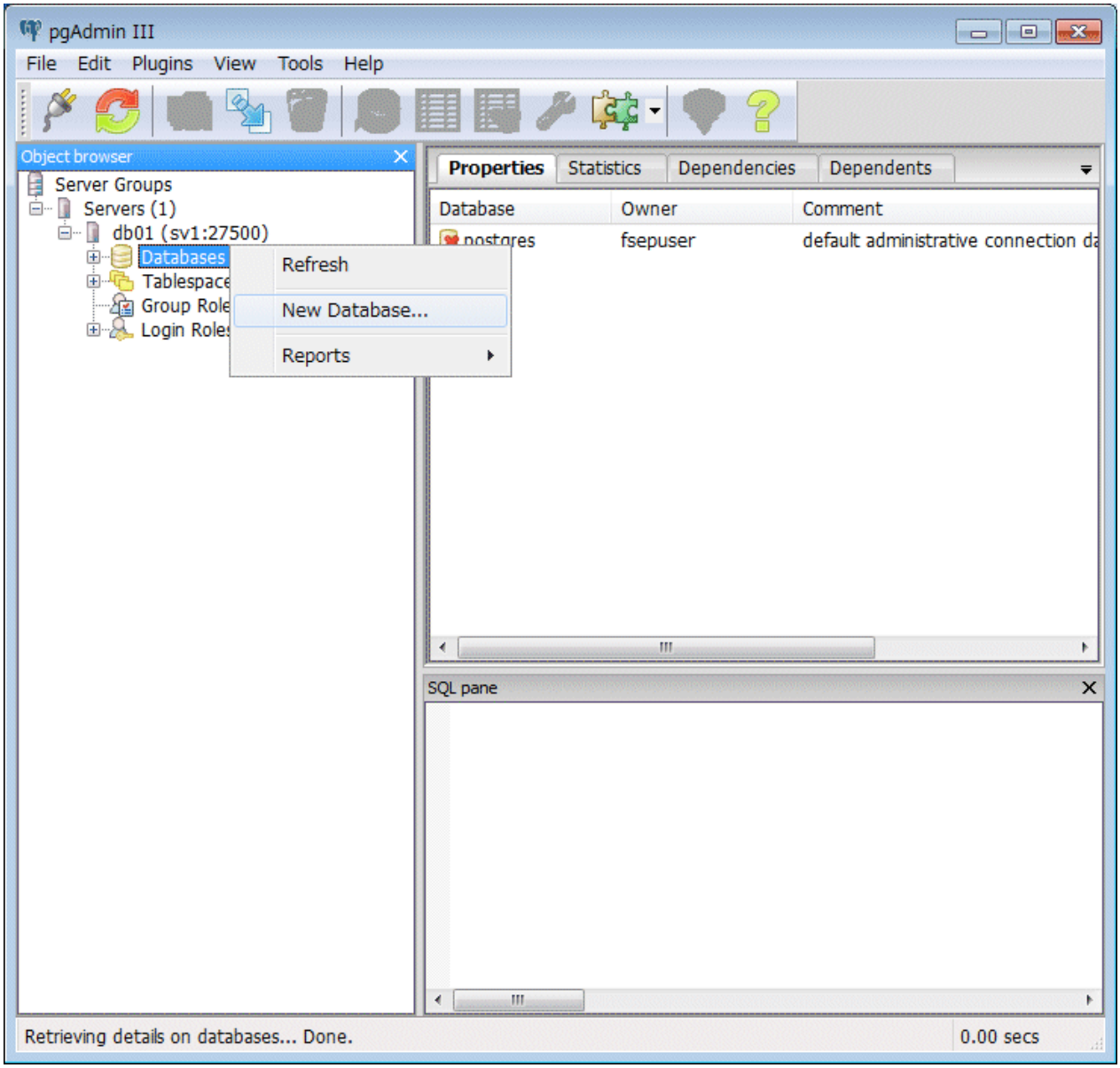
This section explains how to create a database.

- 2.2.1 Using pgAdmin
- 2.2.2 Using Client Commands

2.2.1 Using pgAdmin

Follow the procedure below to define a database using pgAdmin.

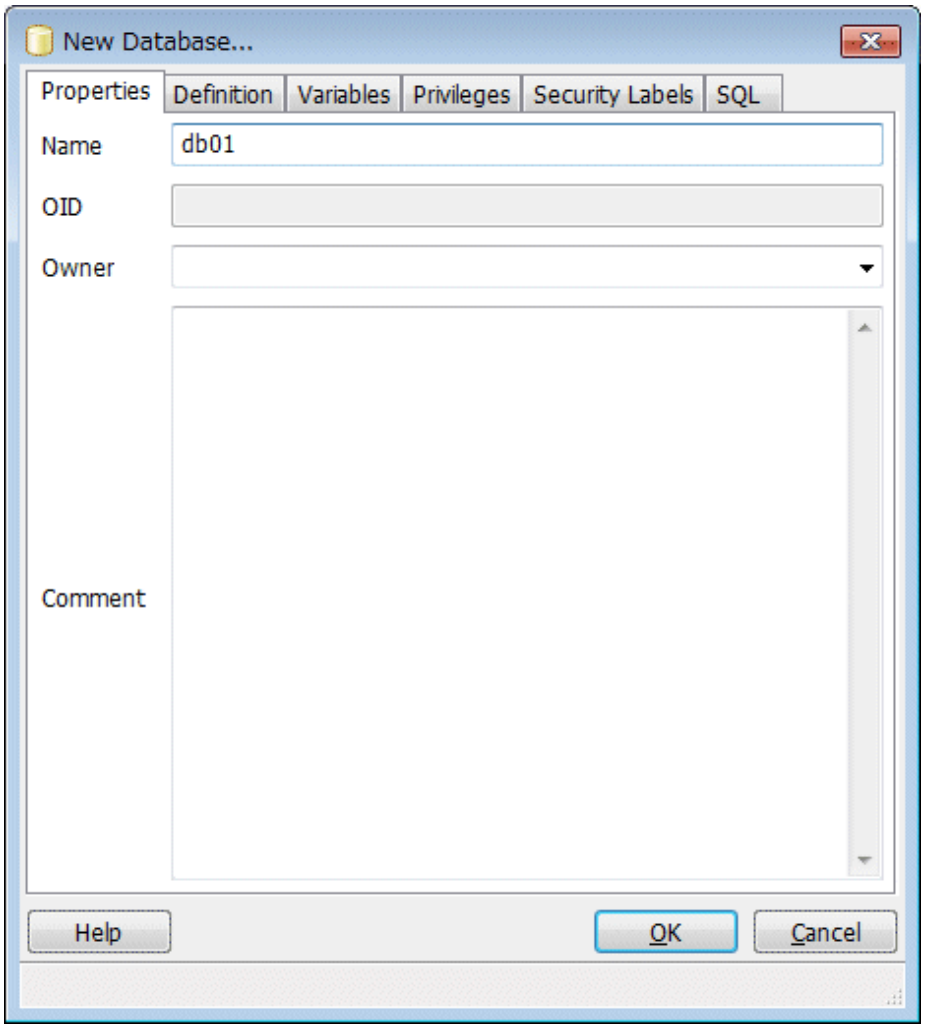
1. In the pgAdmin window, right-click [Database] in [Object browser], and then click [New Database] to display a new database window.



2. Specify appropriate values for the following items in the new database window.

   - [Properties] tab

     The following example illustrates creation of the database "db01".



   - [Name]: Name of the database to be managed

3. Click [OK] to create the database.

2.2.2 Using Client Commands

Follow the procedure below to define a database using client commands.

An example of operations on the server is shown below.

1. Use psql command to connect to the postgres database.
   Execute psql postgres.

```
> psql postgres
psql (9.4.3)
Type "help" for help.
```

2. Create the database.
   To create the database, execute the CREATE DATABASE databaseName; statement.

```
postgres=# CREATE DATABASE db01;
CREATE DATABASE
```

3. Confirm that the database is created.
   Execute the \l+ command, and confirm that the name of the database created in step 2 is displayed.

```
postgres=# \l+
```

4. Disconnect from the postgres database.
   Execute \q to terminate the psql command.

```
postgres=# \q
```

You can create a database using the createdb command.

See

Refer to "Creating a Database" in the "Tutorial" in the PostgreSQL Documentation for information on creating a database using the createdb command.

Chapter 3 Backing Up the Database

This chapter describes how to back up the database.

Backup methods

The following backup methods enable you to recover data to a backup point or to the state immediately preceding disk physical breakdown or data logical failure.

- Backup using WebAdmin

  This method enables you to back up data through intuitive window operations using the GUI.

  WebAdmin is used for recovery.

- Backup using the pgx_dmpall command

  Execute the pgx_dmpall command with a script to perform automatic backup.

  To back up data automatically, you must register the process in the automation software of the operating system. Follow the procedure given in the documentation for your operating system.

  The pgx_rcvall command is used for recovery.

Approximate backup time

The formula for deriving the approximate backup time when you use WebAdmin or the pgx_dmpall command is as follows:

```
backupTime = dataStorageDestinationUsage / diskWritePerformance x 1.5
```

- *dataStorageDestinationUsage*: Disk usage at the data storage destination
- *diskWritePerformance*: Maximum data volume (bytes/second) that can be written per second in the system environment where operation is performed
- 1.5: Coefficient to factor in tasks other than disk write (which is the most time-consuming step)

Note

- Use the selected backup method continuously.

  There are several differences, such as the data format, across the backup methods. For this reason, the following restrictions apply:

  - It is not possible to use one method for backup and another for recovery.
  - It is not possible to convert one type of backup data to a different type of backup data.

- There are several considerations for the backup of the keystore and backup of the database in case the data stored in the database is encrypted. Refer to the following for details:

  - 5.6.4 Backing Up and Recovering the Keystore
  - 5.7 Backing Up and Restoring/Recovering the Database

- If you have defined a tablespace, back it up. If you do not back it up, directories for the tablespace are not created during recovery, which may cause the recovery to fail. If the recovery fails, refer to the system log, create the tablespace, and then perform the recovery process again.

Information

The following methods can also be used to perform backup. Performing a backup using these methods allows you to restore to the point when the backup was performed.

- Backup using an SQL-based dump

  Dump the data by using SQL. This backup method also enables data migration.

- File system level backup

  This backup method requires you to stop the instance and use OS commands to backup database resources as files.

- Backup by continuous archiving

  This is the standard backup method for PostgreSQL.

Refer to "Backup and Restore" in "Server Administration" in the PostgreSQL Documentation for information on these backup methods.

3.1 Periodic Backup

It is recommended that you perform backup periodically.

Backing up data periodically using WebAdmin or the pgx_dmpall command has the following advantages:

- This method reduces disk usage, because obsolete archive logs (transaction logs copied to the backup data storage destination) are deleted. It also minimizes the recovery time when an error occurs.

Backup cycle

The time interval when backup is performed periodically is called the backup cycle. For example, if backup is performed every morning, the backup cycle is 1 day.
The backup cycle depends on the jobs being run, but on Enterprise Postgres it is recommended that operations are run with a backup cycle of at least once per day.

3.2 Backup Methods

This section describes the methods for backing up the database.

- 3.2.1 Using WebAdmin
- 3.2.2 Using Server Commands

3.2.1 Using WebAdmin

You can use WebAdmin to perform backup and check the backup status.

Note

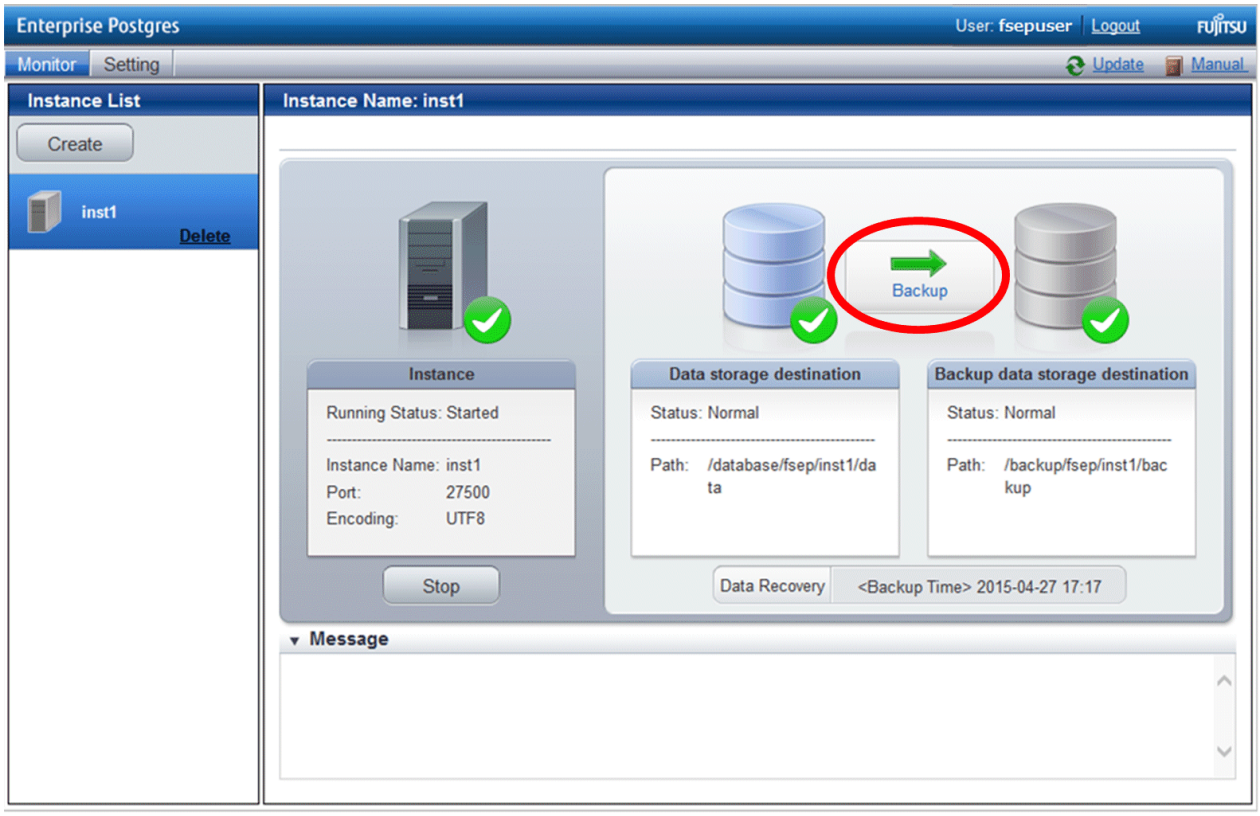
If the data to be stored in the database is to be encrypted, it is necessary to enable the automatic opening of the keystore before doing so. Refer to "5.6.3 Enabling Automatic Opening of the Keystore" for details.

Backup operation

Follow the procedure below to back up the database.

1. Select database backup

   In the [Monitor] window of WebAdmin, click [->] marked "Backup".



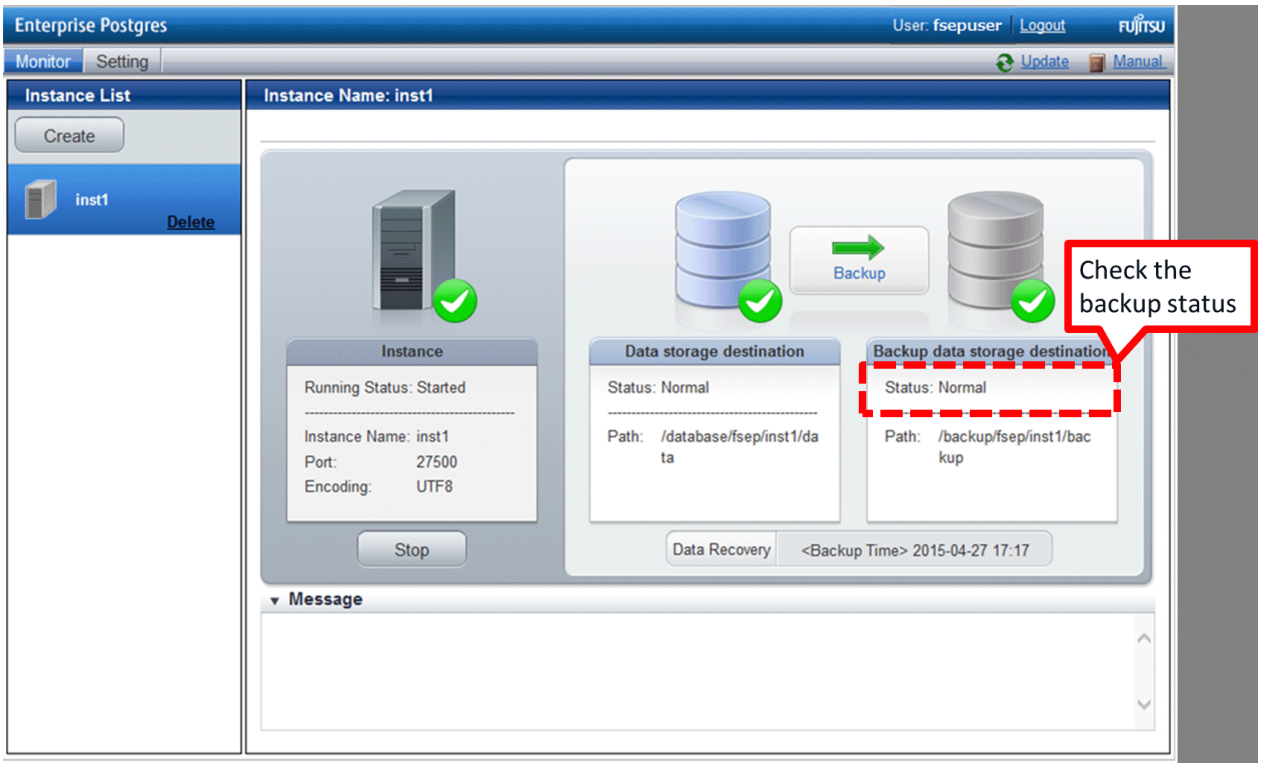
2. Back up the database

   The [Backup] dialog box is displayed. To perform backup, click [Run].
   An instance is automatically started when backup is performed.

Backup status

If an error occurs and backup fails, [Error] is displayed adjacent to [Status] under [Data storage destination] or [Backup data storage destination] in the [Monitor] window. An error message is also displayed in the message list.

In this case, the backup data is not optimized. Ensure that you check the backup result whenever you perform backup. If backup fails, [Solution] appears to the right of the error message. Clicking this button displays information explaining how to resolve the cause of the error. Remove the cause of failure, and perform backup again.



Note

If the data to be stored in the database is to be encrypted, it is necessary to enable the automatic opening of the keystore before doing so. Refer to "5.6.3 Enabling Automatic Opening of the Keystore" for details.

3.2.2 Using Server Commands

Use the pgx_dmpall command and pgx_rcvall command to perform backup and check the backup result.

Preparing for backup

You must prepare for backup before actually starting the backup process.

Follow the procedure below.

See

Refer to " Preparing Directories to Deploy Resources" in the Installation and Setup Guide for Server for information on the location of directories required for backup and for points to take into account.

1. Prepare the backup data storage disk

   For backup, prepare a separate disk unit from the database storage disk and mount it using the operating system commands.

2. Create a directory where the backup data will be stored

   Create an empty directory.

   Set appropriate permissions so that only the instance administrator can access the directory.

   Example

```
# mkdir /backup/inst1
# chown fsepuser:fsepuser /backup/inst1
# chmod 700 /backup/inst1
```

3. Specify the settings required for backup

   Stop the instance, and set the following parameters in the postgresql.conf file.

   Start the instance after editing the postgresql.conf file.

Parameter name	Setting	Description
backup_destination	Name of the directory where the backup data will be stored	Specify the name of the directory where the backup data will be stored. Appropriate privileges that allow only the instance administrator to access the directory must already be set. Place the backup data storage destination directory outside the data storage destination directory, the tablespace directory, and the transaction log storage destination directory.
wal_level	archive or hot_standby(*1)	Specify the output level for the transaction log. *1: hot_standby is a setting for streaming replication.
archive_mode	on	Specify the archive log mode. Specify [on] (execute).
archive_command	'*installationDirectory*/bin/pgx_xlogcopy.cmd "%p" "*backupDataStorageDestinationDirectory*/archived_xlog/%f"'	Specify the path name of the command that will save the transaction log and the storage destination.

   Refer to "Appendix A Parameters" and "Write Ahead Log" under "Server Administration" in the PostgreSQL Documentation for information on the parameters.

Backup operation

Use the pgx_dmpall command to perform backup. You can even embed the pgx_dmpall command in OS automation software to perform backup.

The backup data is stored in the directory specified in the backup_destination parameter of postgresql.conf.

Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

Example

```
> pgx_dmpall -D /database/inst1
```

Note

Backup stores the data obtained during the backup and the backup data of the data obtained during previous backup.

If the data to be stored in the database is encrypted, refer to the following and back up the keystore:

- 5.6.4 Backing Up and Recovering the Keystore

Backup status

Use the pgx_rcvall command to check the backup status.

Specify the following values in the pgx_rcvall command:

- The -l option indicates backup data information.
- Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

```
> pgx_rcvall -l -D /database/inst1
Date                    Status          Dir
2015-05-01 13:30:40     COMPLETE        /backup/inst1/2015-05-01_13-30-40
```

If an error occurs and backup fails, a message is output to the system log.

In this case, the backup data is not optimized. Ensure that you check the backup result whenever you perform backup. If backup fails, remove the cause of failure and perform backup again.

See

Refer to "pgx_dmpall" and "pgx_rcvall" in the Reference for information on the pgx_dmpall command and pgx_rcvall command.

Setting a restore point

In case you want to recover your database to a certain point in time, you can name this particular point in time, which is referred to as the restore point, by using the psql command.

By setting a restore point before executing an application, it becomes easy to identify up to which point in time the data will be reverted.

A restore point can be set to any point in time after a backup is executed. However, if a restore point is set before a backup is executed, the database cannot be recovered to that point in time. This is because restore points are recorded in the archive logs, and the archive logs are discarded when backups are executed.

Example

The following example uses the psql command to connect to the database and execute the SQL statement to set a restore point.

However, when considering continued compatibility of applications, do not use functions directly in SQL statements. Refer to "Notes on Application Compatibility" in the Application Development Guide for details.

```
postgres=# SELECT pg_create_restore_point('batch_20150503_1');
LOG:  restore point "batch_20150503_1" created at 0/20000E8
STATEMENT:  select pg_create_restore_point('batch_20150503_1');
 pg_create_restore_point
-------------------------
 0/20000E8
(1 row)
```

Refer to "8.3.2 Using the pgx_rcvall Command" for information on using a restore point to recover the database.

Note

- Name restore points so that they are unique within the database. Add the date and time of setting a restore point to distinguish it from other restore points, as shown below:
  - YYMMDD_HHMMSS
    - YYMMDD: Indicates the date
    - HHMMSS: Indicates the time

- There is no way to check restore points you have set. Keep a record in, for example, a file.

See

Refer to "System Administration Functions" under "Functions and Operators" in the PostgreSQL Documentation for information on pg_create_restore_point.

Chapter 4 Configuring Secure Communication Using Secure Sockets Layer

If communication data transferred between a client and a server contains confidential information, encrypting the communication data can protect it against threats, such as eavesdropping on the network.

4.1 Configuring Communication Data Encryption

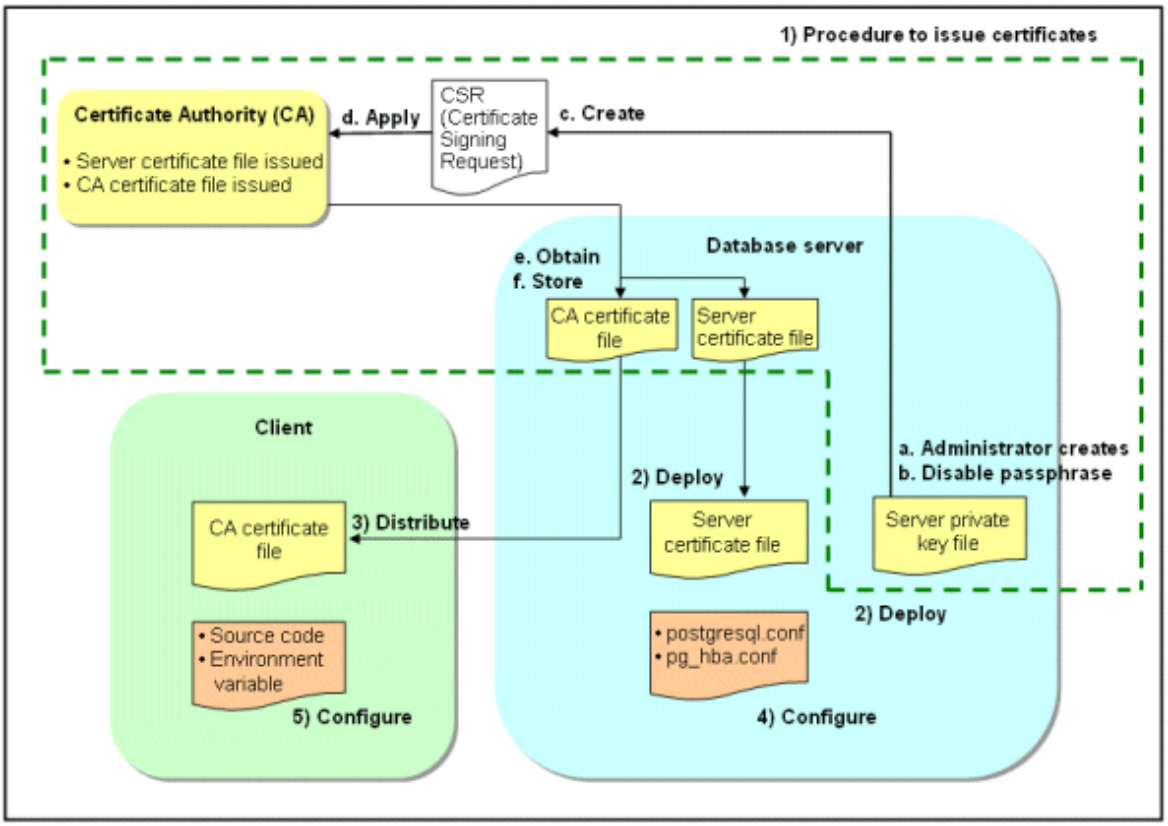
To encrypt communication data transferred between a client and a server, configure communication data encryption as described below. Communication data encryption not only protects the communication content, but it also guards against man-in-the-middle (MITM) attacks (for example, data and password theft through server impersonation).

Table 4.1 Configuration procedure

Configuration procedure
1) Issue a certificate
2) Deploy a server certificate file and a server private key file
3) Distribute a CA certificate file to the client
4) Configure the operating environment for the database server
5) Configure the operating environment for the client

The following figure illustrates the environment for communication data encryption.

Figure 4.1 Environment for communication data encryption



4.1.1 Issuing a Certificate

For authenticating servers, you must acquire a certificate issued by the certificate authority (CA).
Enterprise Postgres supports X.509 standard PEM format files. If the certificate authority issues a file in DER format, use a tool such as the openssl command to convert the DER format file to PEM format.

The following provides an overview of the procedure. Refer to the procedure published by the public or independent certificate authority (CA) that provides the certificate file for details.

a. Create a server private key file

b. Disable the passphrase for the server private key file

c. Create a CSR (signing request for obtaining a server certificate) from the server private key file

d. Apply to the certificate authority (CA) for a server certificate

e. Obtain a server certificate file and a CA certificate file from the certificate authority (CA)

f. Store the server certificate file and the CA certificate file
   Note: If you lose or destroy the certificates, you will need to have them re-issued.

The above procedure enables you to prepare the following files:

- Server private key file
- Server certificate file
- CA certificate file

4.1.2 Deploying a Server Certificate File and a Server Private Key File

Create a directory on the local disk of the database server and store the server certificate file and the server private key file in it.
Use the operating system features to set access privileges for the server certificate file and the server private key file so that only the database administrator has load privileges.
Back up the server certificate file and the server private key file in the event that data corruption occurs and store them securely.

4.1.3 Distributing a CA Certificate File to the Client

Create a directory on the local disk of the client and place the distributed CA certificate file there. Use the operating system features to set load privileges to protect the CA certificate file against accidental deletion.

4.1.4 Configuring the Operating Environment for the Database Server

See

Refer to "Secure TCP/IP Connections with SSL" under "Server Administration" in the PostgreSQL Documentation for details.

4.1.5 Configuring the Operating Environment for the Client

See

Refer to the following sections in the Application Development Guide for details, depending on your application development environment:

- "Settings for Encrypting Communication Data" under "Setup" in "JDBC Driver"
- "Settings for Encrypting Communication Data" under "Setup" in "C Library (libpq)"
- "Settings for Encrypting Communication Data" under "Setup" in "Embedded SQL in C"

4.1.6 Performing Database Multiplexing

When you perform communication that uses database multiplexing and a Secure Socket Layer server certificate, certificates with the same "Common Name" must be used. To ensure this, take one of the following actions:

- Create one server certificate, replicate it, and place a copy on each server used for database multiplexing.
- Create a server certificate with the same "Common Name" for each server used for database multiplexing.

 See

Refer to "Using the Application Connection Switch Feature" in the Application Development Guide for information on how to specify applications on the client.

Chapter 5 Protecting Storage Data Using Transparent Data Encryption

This chapter describes how to encrypt data to be stored in the database.

5.1 Protecting Data Using Encryption

With PostgreSQL, data in a database is protected from access by unauthorized database users through the use of authentication and access controls. However, the OS file is not protected from attackers who bypass the database server's authentication and access controls.

With Enterprise Postgres, data inside the OS file is encrypted, so valuable information is protected even if the file or disk is stolen.

Data to be stored in a database is encrypted when it is written to the data file, and decrypted when it is read.

This is performed automatically by the instance, so the user and the application need not be aware of key management and encryption or decryption. This process is called TDE (Transparent Data Encryption).

The characteristics of TDE are described below.

Encryption mechanisms

Two-layer encryption key and the keystore

In each tablespace, there is a tablespace encryption key that encrypts and decrypts all the data within. The tablespace encryption key is encrypted by the master encryption key and saved.
Only one master encryption key exists in a database cluster. It is encrypted based on a passphrase specified by the user and stored in a keystore. Enterprise Postgres provides a file-based keystore. Attackers who do not know the passphrase cannot read the master encryption key from the keystore.

Strong encryption algorithms

TDE uses the Advanced Encryption Standard (AES) as its encryption algorithm. AES was adopted as a standard in 2002 by the United States Federal Government, and is used throughout the world.

Faster encryption and decryption based on hardware

TDE minimizes the overhead of encryption and decryption by using the AES-NI (Advanced Encryption Standard New Instructions) built into Intel(R) Xeon(R) processors since the 5600 series. This means that even in situations where previously the minimum encryption target was selected as a tradeoff between performance and security, it is now possible to encrypt all the data of an application.

You can reference a list of processors equipped with AES-NI on the following page at Intel Corporation's website:

http://ark.intel.com/search/advanced/?s=t&AESTech=true

Zero overhead storage areas

Encryption does not change the size of data stored in tables, indexes, or WAL. There is, therefore, no need for additional estimates or disks.

Scope of encryption

All user data within the specified tablespace

The tablespace is the unit for specifying encryption. All tables, indexes, temporary tables, and temporary indexes created in the encrypted tablespace are encrypted. There is no need for the user to consider which tables and strings to encrypt.

Backup data

The pgx_dmpall command and pg_basebackup command create backup data by copying the OS file. Backups of the encrypted data are, therefore, also encrypted. Information is protected from leakage even if the backup medium is stolen.

WAL and temporary files

WAL, which is created by updating encrypted tables and indexes, is encrypted with the same security strength as the update target. When large merges and sorts are performed, the encrypted data is written to a temporary file in encrypted format.

Streaming replication support

You can combine streaming replication and transparent data encryption. The data and WAL encrypted on the primary server is transferred to the standby server in its encrypted format and stored.

Note

The following are not encrypted:

- pg_dump and pg_dumpall output files
- Files output by the COPY command
- Notification event payloads that communicate using the LISTEN or NOTIFY command

5.2 Setting the Master Encryption Key

To use transparent data encryption, you must create a keystore and set the master encryption key.

1. In the keystore_location parameter of postgresql.conf, specify the directory to store the keystore.

   Specify a different location for each database cluster.

   ```
   keystore_location = '/key/store/location'
   ```

   Refer to "Appendix A Parameters" for information on postgresql.conf.

   After editing the postgresql.conf file, either start or restart the instance.

   - Using WebAdmin

     Refer to "2.1.1 Using WebAdmin", and restart the instance.

   - Using the pg_ctl command

     Specify the following in the pg_ctl command:

     - Specify "restart" as the mode.
     - Specify the data storage destination directory in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
     - Specify the -w option. This means that the command returns after waiting for the instance to start. If the -w option is not specified, it may not be possible to determine if the starting of the instance completed successfully or if it failed.

     Example

     ```
     > pg_ctl restart -w -D /database/inst1
     ```

2. Execute an SQL function, such as the one below, to set the master encryption key. This must be performed by the superuser. Execute it as the database superuser.

   ```
   SELECT pgx_set_master_key('passphrase');
   ```

   The value "passphrase" is the passphrase that will be used to open the keystore. The master encryption key is protected by this passphrase, so avoid specifying a short simple string that is easy to guess.

   Refer to "B.2 Transparent Data Encryption Control Functions" for information on the pgx_set_master_key function.

Note

Note that if you forget the passphrase, you will not be able to access the encrypted data. There is no method to retrieve a forgotten passphrase and decrypt data. Do not, under any circumstances, forget the passphrase.

The pgx_set_master_key function creates a file with the name keystore.ks in the keystore storage destination. It also creates a master encryption key from random bit strings, encrypts it with the specified passphrase, and stores it in keystore.ks. At this point, the keystore is open.

5.3 Opening the Keystore

To create encrypted tablespaces and access the encrypted data, you must first open the keystore. When you open the keystore, the master encryption key is loaded into the database server memory and becomes usable for encryption and decryption.

You need to open the keystore each time you start the instance. To open the keystore, the database superuser must execute the following SQL function.

```
SELECT pgx_open_keystore('passphrase');
```

The value "passphrase" is the passphrase specified during creation of the keystore.

Refer to "B.2 Transparent Data Encryption Control Functions" for information on the pgx_open_keystore function.

Note that, in the following cases, the passphrase must be entered when starting the instance, because the encrypted WAL must be decrypted for recovery. In this case, the above-mentioned pgx_open_keystore function cannot be executed.

- If performing crash recovery at the time of starting the instance
- If performing recovery using continuous archiving

For the above cases, specify the --keystore-passphrase option in the pg_ctl command, and then start the instance. This will display the prompt for the passphrase to be entered, as shown below.

```
> pg_ctl --keystore-passphrase start
Enter the passphrase:
The server is starting
>
```

Point

When using an automatically opening keystore, you do not need to enter the passphrase and you can automatically open the keystore when the database server starts. Refer to "5.6.3 Enabling Automatic Opening of the Keystore" for details.

5.4 Encrypting a Tablespace

The keystore must be open before you can create an encrypted tablespace.

When creating a tablespace that will be encrypted, configure the encryption algorithm in the runtime parameters. For example, to create a tablespace with the name secure_tablespace using AES with a key length of 256 bits as the encryption algorithm, configure as shown below.

```
-- Specify the encryption algorithm for the tablespace to be created below
SET tablespace_encryption_algorithm = 'AES256';
CREATE TABLESPACE secure_tablespace LOCATION '/My/Data/Dir';
-- Specify that the tablespace to be created below is not to be encrypted
SET tablespace_encryption_algorithm = 'none';
```

Or

```
CREATE TABLESPACE secure_tablespace LOCATION '/My/Data/Dir' tablespace_encryption_algorithm =
'AES256';
```

You can use AES with a key length of 128 bits or 256 bits as the encryption algorithm. It is recommended that you use 256-bit AES. Refer to "Appendix A Parameters" for information on how to specify the runtime parameters.

If user provides both GUC and command line options while creating the tablespace, the preference is given to the command line option.

The pg_default and pg_global tablespaces cannot be encrypted.

Create tables and indexes in the encrypted tablespace that you created. Relations created in the encrypted tablespace are automatically encrypted.

Example

Example 1: Specifying an encrypted tablespace when creating it

```
CREATE TABLE my_table (...)
    TABLESPACE secure_tablespace;
```

Example 2: Not explicitly specifying a tablespace when creating it and instead using the default tablespace

```
SET default_tablespace = 'secure_tablespace';
CREATE TABLE my_table (...);
```

The process is the same for encrypting temporary tables and temporary indexes. In other words, either explicitly specify the TABLESPACE clause or list encrypted tablespaces in the temp_tablespaces parameter, and then execute CREATE TEMPORARY TABLE or CREATE INDEX.

If you specify an encrypted tablespace in the TABLESPACE clause of the CREATE DATABASE statement when creating a database, relations that you create in the database without explicitly specifying a tablespace will be encrypted. Furthermore, the system catalog is also encrypted, so the source code of user-defined functions is also protected.

Note

An encrypted tablespace cannot be created from the window used for creating the pgAdmin tablespace, or from the query tool. To create an encrypted tablespace, click [PSQL Console] from the [Plugins] menu and create an encrypted tablespace in the psql console window.

5.5 Checking an Encrypted Tablespace

The pgx_tablespaces system view displays information about whether each tablespace has been encrypted, and about the encryption algorithm. Refer to "C.1 pgx_tablespaces" for information on strings.

You can discover which tablespaces have been encrypted by executing the following SQL statements.

However, when considering continued compatibility of applications, do not reference system catalogs (pg_tablespace) directly in SQL statements.

```
SELECT spcname, spcencalgo
FROM pg_tablespace ts, pgx_tablespaces tsx
WHERE ts.oid = tsx.spctablespace;
```

Example

```
postgres=# SELECT spcname, spcencalgo FROM pg_tablespace ts, pgx_tablespaces tsx WHERE ts.oid =
tsx.spctablespace;
      spcname      | spcencalgo
-------------------+------------
 pg_default        | none
 pg_global         | none
 secure_tablespace | AES256
(3 rows)
```

See

Refer to "Notes on Application Compatibility" in the Application Development Guide for information on how to maintain application compatibility.

5.6 Managing the Keystore

This section describes how to manage the keystore and the master encryption key to guard against the threat of theft.

5.6.1 Changing the Master Encryption Key

Using the same encryption key for an extended period gives attackers an opportunity to decipher the encrypted data. It is recommended that you change the key at regular intervals, or whenever the key is exposed to risk.

Adhere to the industry's best practices for encryption algorithms and key management when considering how often the key should be changed. For example, the NIST in the United States has published "NIST Special Publication 800-57". The PCI DSS also refers to this publication. This publication recommends changing the master encryption key once a year.

To change the master encryption key, execute the pgx_set_master_key function, which is the same function used for configuring the key. Refer to "5.2 Setting the Master Encryption Key" for details.

After changing the master encryption key, you must immediately back up the keystore.

5.6.2 Changing the Keystore Passphrase

In security policies for organizations, it is usually a requirement that the passphrase be changed whenever a security administrator who knows the passphrase is removed from duties due to transfer or retirement. It is also recommended that the passphrase be changed if it is ever exposed to risks due to deception such as social engineering.

To change the keystore passphrase, execute the following SQL function as a superuser.

```
SELECT pgx_set_keystore_passphrase('oldPassphrase', 'newPassphrase');
```

After changing the passphrase, you must immediately back up the keystore.

Refer to "B.2 Transparent Data Encryption Control Functions" for information on the pgx_set_keystore_passphrase function.

5.6.3 Enabling Automatic Opening of the Keystore

When using an automatically opening keystore, you do not need to enter the passphrase and you can automatically open the keystore when the instance starts. Execute the pgx_keystore command to enable automatic opening of the keystore.

```
> pgx_keystore --enable-auto-open /key/store/location/keystore.ks
Enter the passphrase:
Automatic opening of the keystore is now enabled
>
```

See

Refer to "pgx_keystore" in the Reference for information on pgx_keystore command.

When automatic opening is enabled, an automatically opening keystore is created in the same directory as the original keystore. The file name of the automatically opening keystore is keystore.aks. The file keystore.aks is an obfuscated copy of the decrypted content of the keystore.ks file. As long as this file exists, there is no need to enter the passphrase to open the keystore when starting the instance.

Do not delete the original keystore file, keystore.ks. It is required for changing the master encryption key and the passphrase. When you change the master encryption key and the passphrase, keystore.aks is recreated from the original keystore file, keystore.ks.

Protect keystore.ks, keystore.aks, and the directory that stores the keystore so that only the user who starts the instance can access them.

Configure the permission of the files so that only the user who starts the instance can access the SQL functions and commands that create these files. Accordingly, manually configure the same permission mode if the files are restored.

Example

```
# chown -R fsepuser:fsepuser /key/store/location
# chmod 700 /key/store/location
# chmod 600 /key/store/location/keystore.ks
# chmod 600 /key/store/location/keystore.aks
```

An automatically opening keystore will only open on the computer where it was created.

To disable automatic opening of the keystore, delete keystore.aks.

Note

- To use WebAdmin for recovery, you must enable automatic opening of the keystore.
- Refer to "5.7 Backing Up and Restoring/Recovering the Database" after enabling or reconfiguring encryption to back up the database.
- Specify a different directory from those below as the keystore storage destination:
  - Data storage destination
  - Tablespace storage destination
  - Transaction log storage destination
  - Backup data storage destination

5.6.4 Backing Up and Recovering the Keystore

Back up the keystore at the following times in case it is corrupted or lost. Note that you must store the database and the keystore on separate data storage media. Storing both on the same data storage medium risks the danger of the encrypted data being deciphered if the medium is stolen. A passphrase is not required to open an automatically opening keystore, so store this type of keystore in a safe location.

- When the master encryption key is first configured
- When the master encryption key is changed
- When the database is backed up
- When the keystore passphrase is changed

Point

Do not overwrite an old keystore when backing up a keystore. This is because during database recovery, you must restore the keystore to its state at the time of database backup. When the backup data of the database is no longer required, delete the corresponding keystore.

Example

- Back up the database and the keystore on May 1, 2015.

```
> pgx_dmpall -D /database/inst1
> cp -p /key/store/location/keystore.ks /keybackup/keystore_20150501.ks
```

  Specify the following in the pgx_dmpall command:

  - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

- Change the master encryption key, and back up the keystore on May 5, 2015.

```
> psql -c "SELECT pgx_set_master_key('passphrase')" postgres
> cp -p /key/store/location/keystore.ks /keybackup/keystore_20150505.ks
```

  Specify the following in the psql command:

  - Specify the SQL function that sets the master encryption key in the -c option.
  - Specify the name of the database to be connected to as the argument.

If the keystore is corrupted or lost, restore the keystore containing the latest master encryption key. If there is no keystore containing the latest master encryption key, restore the keystore to its state at the time of database backup, and recover the database from the database backup. This action recovers the keystore to its latest state.

Example

- Restore the keystore containing the latest master encryption key as of May 5, 2015.

```
> cp -p /keybackup/keystore_20150505.ks /key/store/location/keystore.ks
```

- If there is no backup of the keystore containing the latest master encryption key, recover the keystore by restoring the keystore that was backed up along with the database on 1 May 2015.

```
> cp -p /keybackup/keystore_20150501.ks /key/store/location/keystore.ks
> pgx_rcvall -B /backup/inst1 -D /database/inst1 --keystore-passphrase
```

  Specify the following in the pgx_rcvall command:

  - Specify the data storage directory in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
  - Specify the backup data storage directory in the -B option.
  - The --keystore-passphrase option prompts you to enter the passphrase to open the keystore.

If you have restored the keystore, repeat the process of enabling automatic opening of the keystore. This ensures that the contents of the automatically opening keystore (keystore.aks) are identical to the contents of the restored keystore.

It is recommended that you do not back up the automatically opening keystore file, keystore.aks. If the database backup medium and the backup medium storing the automatically opening keystore are both stolen, the attacker will be able to read the data even without knowing the passphrase.

If the automatically opening keystore is corrupted or lost, you must again enable automatic opening. The keystore.aks file will be recreated from keystore.ks at this time.

See

Refer to "pgx_rcvall" and "pgx_dmpall" in the Reference for information on the pgx_rcvall and pgx_dmpall commands.

Refer to "psql" under "Reference" in the PostgreSQL Documentation for information on the psql command.

Refer to "B.2 Transparent Data Encryption Control Functions" for information on the pgx_set_master_key function.

Refer to "5.6.3 Enabling Automatic Opening of the Keystore" for information on how to enable automatic opening of the keystore.

5.7 Backing Up and Restoring/Recovering the Database

Enterprise Postgres enables you to use the five backup and recovery methods described below. Regardless of the method you use, you must back up the keystore at the same time.

Note that you must store the database and the keystore on separate data storage media. Storing both on the same data storage medium risks the danger of the encrypted data being deciphered if the medium is stolen.

Backup and recovery using WebAdmin

- Backup

  WebAdmin backs up encrypted data.

  Back up the key store after backing up the database.

- Recovery

  Restore the keystore to its state at the time of database backup. Refer to "5.6.4 Backing Up and Recovering the Keystore" for details.

  Enable automatic opening of the keystore in accordance with the procedure described in "5.6.3 Enabling Automatic Opening of the Keystore". Then, use WebAdmin to recover the database.

Backup and recovery using the pgx_dmpall and pgx_rcvall commands

- Backup

  The pgx_dmpall command backs up the encrypted data.

  Back up the key store after backing up the database.

- Recovery

  Restore the keystore to its state at the time of the database backup.

  Configure automatic opening of the key store as necessary.

  If automatic opening of the keystore is not enabled, execute the pgx_rcvall command with the --keystore-passphrase option specified. This will display the prompt for the passphrase to be entered.

Example

- Back up the database and the keystore on May 1, 2015.

```
> pgx_dmpall -D /database/inst1
> cp -p /key/store/location/keystore.ks /keybackup/keystore_20150501.ks
```

  Specify the following in the pgx_dmpall command:

  - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

- Recover the database and the keystore from the backup taken on May 1, 2015.

```
> cp -p /keybackup/keystore_20150501.ks /key/store/location/keystore.ks
> pgx_keystore --enable-auto-open /key/store/location/keystore.ks  (Execute only when enabling
automatic opening)
> pgx_rcvall -B /backup/inst1 -D /database/inst1 --keystore-passphrase
```

  Specify the following in the pgx_rcvall command:

  - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
  - Specify the backup data storage directory in the -B option.
  - The --keystore-passphrase option prompts you to enter the passphrase to open the keystore.

Dump and restore using SQL

- Backup

  The files output by the pg_dump and pg_dumpall commands are not encrypted. You should, therefore, encrypt the files using OpenSSL commands or other means before saving them, as described in "5.8 Importing and Exporting the Database" below.

  Back up the key store after backing up the database.

- Restore

  If the backup data has been encrypted using, for example Open SSL commands, decrypt that data.

  The data generated by the pg_dumpall command includes a specification to encrypt tablespaces by For this reason, the pg_restore command encrypts tablespaces during restoration.

File system level backup and restore

- Backup

  Stop the instance and backup the data directory and the tablespace directory using the file copy command of the operating system. The files of encrypted tablespaces are backed up in the encrypted state.

  Back up the key store after performing the backup.

- Restore

  Restore the keystore to its state at the time of the database backup.

  Stop the instance and restore the data directory and the tablespace directory using the file copy command of the operating system.

Continuous archiving and point-in-time recovery

- Backup

  The pg_basebackup command backs up the encrypted data as is.

  Back up the key store after performing the backup.

- Recovery

  Restore the keystore to its state at the time of the database backup.

  Configure automatic opening of the key store as necessary.

  If automatic opening of the keystore is not enabled, execute the pg_ctl command to start the instance with the --keystore-passphrase option specified. This will display the prompt for the passphrase to be entered.

See

- Refer to "pg_ctl" under "Reference" in the PostgreSQL Documentation for information on the pg_ctl command.
- Refer to "Reference" in the PostgreSQL Documentation for information on the following commands:
  - psql
  - pg_dump
  - pg_restore
  - pg_basebackup
- Refer to the Reference for information on the following commands:
  - pgx_rcvall
  - pgx_dmpall
  - pg_dumpall

If you have restored the keystore, repeat the process of enabling automatic opening of the keystore This ensures that the contents of the automatically opening keystore (keystore.aks) are identical to the contents of the restored keystore.

Refer to "5.6.3 Enabling Automatic Opening of the Keystore" for information on how to enable automatic opening of the keystore.

5.8 Importing and Exporting the Database

The files output by the COPY TO command are not encrypted. Therefore, when transferring files to other systems, you should encrypt files using OpenSSL commands or other means and use scp or sftp to encrypt the data being transferred.

Use a safe method to delete obsolete plain text files.

You can use the following methods to safely delete files:

- shred command

Example

```
# Export the contents of the table my_table to a CSV file.
> psql -c "COPY my_table TO '/tmp/my_table.csv' (FORMAT CSV)" postgres

# Encrypt the exported file.
> openssl enc -e -aes256 -in my_table.csv -out my_table.csv.enc
(The user is prompted to enter the passphrase to be used for encryption)

# Safely delete plain text files.
> shred -u -x my_table.csv
 (Transfer encrypted files to other systems)

# Decrypt the encrypted files on other systems.
> openssl enc -d -aes256 -in my_table.csv.enc -out my_table.csv
(The user is prompted to enter the passphrase to be used for decryption)
```

If you use COPY FROM to import data to tables and indexes in an encrypted tablespace, the imported data is automatically encrypted before being stored.

5.9 Encrypting Existing Data

You cannot encrypt existing unencrypted tablespaces. In addition, you cannot change encrypted tablespaces so that they do not encrypt.

As an alternative, transfer the tables and indexes to other tablespaces. You can use the following SQL commands for this.

```
ALTER TABLE table_name SET TABLESPACE new_tablespace;
ALTER INDEX index_name SET TABLESPACE new_tablespace;
ALTER DATABASE database_name SET TABLESPACE new_tablespace;
```

See

Refer to "SQL Commands" under "Reference" in the PostgreSQL Documentation for information on SQL commands.

5.10 Operations in Cluster Systems

This section describes how to use transparent data encryption on cluster systems such as high-availability systems, streaming replication, C, and the Mirroring Controller option.

5.10.1 HA Clusters that do not Use Database Multiplexing

Take the following points into account when using transparent data encryption in an HA cluster environment that does not use database multiplexing.

Placement and automatic opening of the keystore file

There are two alternatives for placing the keystore file:

- Sharing the keystore file
- Placing a copy of the keystore file

Sharing the keystore file

This involves using the same keystore file on the primary server and the standby server.

As the standby server is not active while the primary server is running, this file would not be accessed simultaneously, and therefore, it can be shared.

To manage the keystore file in a more secure manner, place it on the key management server or the key management storage isolated in a secure location.

Enable the automatic opening of the keystore on both the primary and standby servers.

Placing a copy of the keystore file

This involves placing a copy of the primary server keystore file on the standby server.

You can do this if you cannot prepare a shared server or disk device that can be accessed from both the primary and standby servers.

However, if you change the master encryption key and the passphrase on the primary server, you must copy the keystore file to the standby server again.

To manage the keystore file in a more secure manner, prepare the key management server or the key management storage isolated in a secure location for both the primary and standby servers, and place the keystore files there.

Enable the automatic opening of the keystore on both the primary and standby servers. Note that copying the automatically opening keystore file (keystore.aks) to the standby server does not enable the automatic opening of the keystore.

 See

Refer to the Cluster Operation Guide for information on building a cluster system environment using failover operation.

5.10.2 Database Multiplexing Mode

Note the following when using transparent data encryption in environments that use streaming replication, database multiplexing with streaming replication, or the Mirroring Controller option.

Placing the keystore file

Place a copy of the primary server keystore file on the standby server.

This is required as the keystore file cannot be shared, and both servers may need to access it simultaneously.

 Point

To manage the keystore file in a more secure manner, place it on the key management server or the key management storage isolated in a secure location. A keystore used by both the primary and standby servers can be managed on the same key management server or key management storage.

However, create different directories for the keystores to be used by the primary server and the standby server. Then copy the keystore for the primary server to the directory used on the standby server.

Automatically opening the keystore

You must enable automatic opening of the keystore.

To do this, enable automatic opening of the keystore in all servers that make up database multiplexing. The settings for automatic opening of the keystore include information unique to each server, so simply copying the file does not enable it.

Changing the passphrase

Changes to the passphrase are reflected in all servers that make up database multiplexing, so no special operation is required.

Building and starting a standby server

Before using the pg_basebackup command or pgx_rcvall command to build a standby server, copy the keystore file from the primary server to the standby server. When using an automatically opening keystore, use the copied keystore file to enable automatic opening on the standby server.

Open the keystore each time you start the standby server. This step is necessary for decrypting and restoring encrypted WAL received from the primary server. To open the keystore, specify the --keystore-passphrase option in the pg_ctl command or pgx_rcvall command and enter the passphrase, or use an automatically opening keystore.

Changing the master encryption key and the passphrase

Change the master encryption key and the passphrase on the primary server. You need not copy the keystore from the primary server to the standby server. You need not even restart the standby server or reopen the keystore. Changes to the master encryption key and the passphrase are reflected in the keystore on the standby server.

See

Refer to "pgx_rcvall " in the Reference for information on pgx_rcvall command.

Refer to "pg_ctl" under "Reference" in the PostgreSQL Documentation for information on pg_ctl command.

Refer to "pg_basebackup" under "Reference" in the PostgreSQL Documentation for information on pg_basebackup command.

Refer to "High Availability, Load Balancing, and Replication" under "Server Administration" in the PostgreSQL Documentation for information on how to set up streaming replication.

5.11 Security-Related Notes

- stored in a core file, which is a process memory dump. You should, therefore, safely delete the memory dump.
  You can safely delete files by using the following command:
  - shred command
- Unencrypted data may be written from the database server memory to the operating system's swap area. To prevent leakage of information from the swap area, consider either disabling the use of swap area or encrypting the swap area using a full-disk encryption product.
- The content of the server log file is not encrypted. Therefore, in some cases the value of a constant specified in a SQL statement is output to the server log file. To prevent this, consider setting a parameter such as log_min_error_statement.
- When executing an SQL function that opens the keystore and modifies the master encryption key, ensure that the SQL statement containing the passphrase is not output to the server log file. To prevent this, consider setting a parameter such as log_min_error_statement. If you are executing this type of SQL function on a different computer from the database server, encrypt the communication between the client and the database server with SSL.

5.12 Tips for Installing Built Applications

With transparent data encryption, you can easily encrypt all the data in an application without modifying the application. Database administrators install built applications in the following manner. However, this procedure stores data to the default tablespace, so take necessary action if processing differs from the original design.

1. (Normal procedure) Create an owner and a database for the built application.

```
CREATE USER crm_admin ...;
CREATE DATABASE crm_db ...;
```

2. (Procedure for encryption) Create an encrypted tablespace to store the data for the built application.

```
SET tablespace_encryption_algorithm = 'AES256';
CREATE TABLESPACE crm_tablespace LOCATION '/crm/data';
```

3. (Procedure for encryption) Configure an encrypted tablespace as the default tablespace for the owner of the built application.

```
ALTER USER crm_admin SET default_tablespace = 'crm_tablespace';
ALTER USER crm_admin SET temp_tablespaces = 'crm_tablespace';
```

4. (Normal procedure) Install the built application. The application installer prompts you to enter the host name and the port number of the database server, the user name, and the database name. The installer uses the entered information to connect to the database server and execute the SQL script. For applications that do not have an installer, the database administrator must manually execute the SQL script.

Normally, the application's SQL script includes logic definition SQL statements, such as CREATE TABLE, CREATE INDEX, and GRANT or REVOKE, converted from the entity-relationship diagram. It does not include SQL statements that create databases, users, and tablespaces. Configuring the default tablespace of the users who will execute the SQL script deploys the objects generated by the SQL script to the tablespace.

Chapter 6 Periodic Operations

This chapter describes the operations that must be performed periodically when running daily database jobs.

6.1 Configuring and Monitoring the Log

Enterprise Postgres enables you to output database errors and warnings to a log file.

This information is useful for identifying if errors have occurred and the causes of those errors.

By default, this information is output to the system log. It is recommended that you configure Enterprise Postgres to collect logs from its log files (for example, log_destination) before operating Enterprise Postgres.

Periodically monitor the log files to check if any errors have occurred.

See

- Refer to "Error Reporting and Logging" under "Server Administration" in the PostgreSQL Documentation for information on logs.
- Refer to "Configuring Parameters" in the Installation and Setup Guide for Server for information on log settings when operating with WebAdmin.

6.2 Monitoring Disk Usage and Securing Free Space

When a database is used for an extended period, free space on the disk is continuously consumed and in some cases the disk space runs out. When this happens, database jobs may stop and no longer run.

You should, therefore, periodically monitor the usage of disk space, and delete obsolete files located in the disk.

Monitor the disk usage of the disk where the following directories are located:

- Data storage destination directory
- Transaction log storage destination (if the transaction log is stored in a different directory from the data storage destination directory)
- Backup data storage destination directory
- Tablespace storage destination directory

6.2.1 Monitoring Disk Usage

To check the disk usage, use the following operating system commands:

- df command

You can even use SQL statements to check tables and indexes individually.

Refer to "Determining Disk Usage" under "Server Administration" in the PostgreSQL Documentation for information on this method.

Information

If you are using WebAdmin for operations, a warning is displayed when disk usage reaches 80%

6.2.2 Securing Free Disk Space

Secure free disk space by using the following operating system commands to delete unnecessary files, other than the database, from the same disk unit.

- rm command

You can also secure disk space by performing the following tasks periodically:

- To secure space on the data storage destination disk:

  Execute the REINDEX statement. Refer to "6.5 Reorganizing Indexes" for details.

- To secure space on the backup data storage destination disk:

  Execute backup using WebAdmin or the pgx_dmpall command.

6.3 Automatically Closing Connections

If an application stops responding and abnormally terminates for any reason, the connection from the application may remain active on the database server. If this situation continues for an extended period, other applications attempting to connect to the database server may encounter an error, or an error indicating that the tables are unavailable may occur.

It is, therefore, recommended that idle connections be closed automatically at regular intervals.

Set the following parameters in the postgresql.conf file to indicate the time permitted to elapse before a connection is closed.

Parameter name	Setting	Description
tcp_keepalives_idle	Time until keepalive is sent (seconds) If 0, the default value of the system is used.	Sends keepalive to an idle connection at the specified interval in seconds It is recommended to specify 30 seconds.
tcp_keepalives_interval	keepalive send interval (seconds) If 0, the default value of the system is used.	Sends keepalive at the specified interval It is recommended to specify 6 seconds.

See

Refer to "Connection Settings" under "Server Administration" in the PostgreSQL Documentation for information on the parameters.

6.4 Monitoring the Connection State of an Application

Enterprise Postgres does not immediately delete the updated or deleted data. If the VACUUM determines there are no transactions that reference the database, Enterprise Postgres collects obsolete data.

However, obsolete data is not collected if there are connections that have remained active for an extended period or connections occupying resources. In this case the database may expand, causing performance degradation.

See

Refer to "Routine Vacuuming" under "Server Administration" in the PostgreSQL Documentation for information on the VACUUM command.

In such cases, you can minimize performance degradation of the database by monitoring problematic connections.

The following two methods are supported for monitoring connections that have been in the waiting status for an extended period:

- 6.4.1 Using the View (pg_stat_activity)
- 6.4.2 Using pgAdmin

6.4.1 Using the View (pg_stat_activity)

Use the view (pg_stat_activity) to identify and monitor connections where the client has been in the waiting status for an extended period.

Example

The example below shows connections where the client has been in the waiting status for at least 60 minutes.

However, when considering continued compatibility of applications, do not reference system catalogs directly in the following SQL statements.

```
postgres=# select * from pg_stat_activity where state='idle in transaction' and current_timestamp >
cast(query_start + interval '60 minutes' as timestamp);
-[ RECORD 1 ]----+------------------------------
datid            | 13003
datname          | db01
pid              | 4638
usesysid         | 10
usename          | fsep
application_name | apl01
client_addr      | 192.33.44.15
client_hostname  |
client_port      | 27500
backend_start    | 2015-04-24 09:09:21.730641+09
xact_start       | 2015-04-24 09:09:23.858727+09
query_start      | 2015-04-24 09:09:23.858727+09
state_change     | 2015-04-24 09:09:23.858834+09
waiting          | f
state            | idle in transaction
backend_xid      |
backend_xmin     |
query            | begin;
```

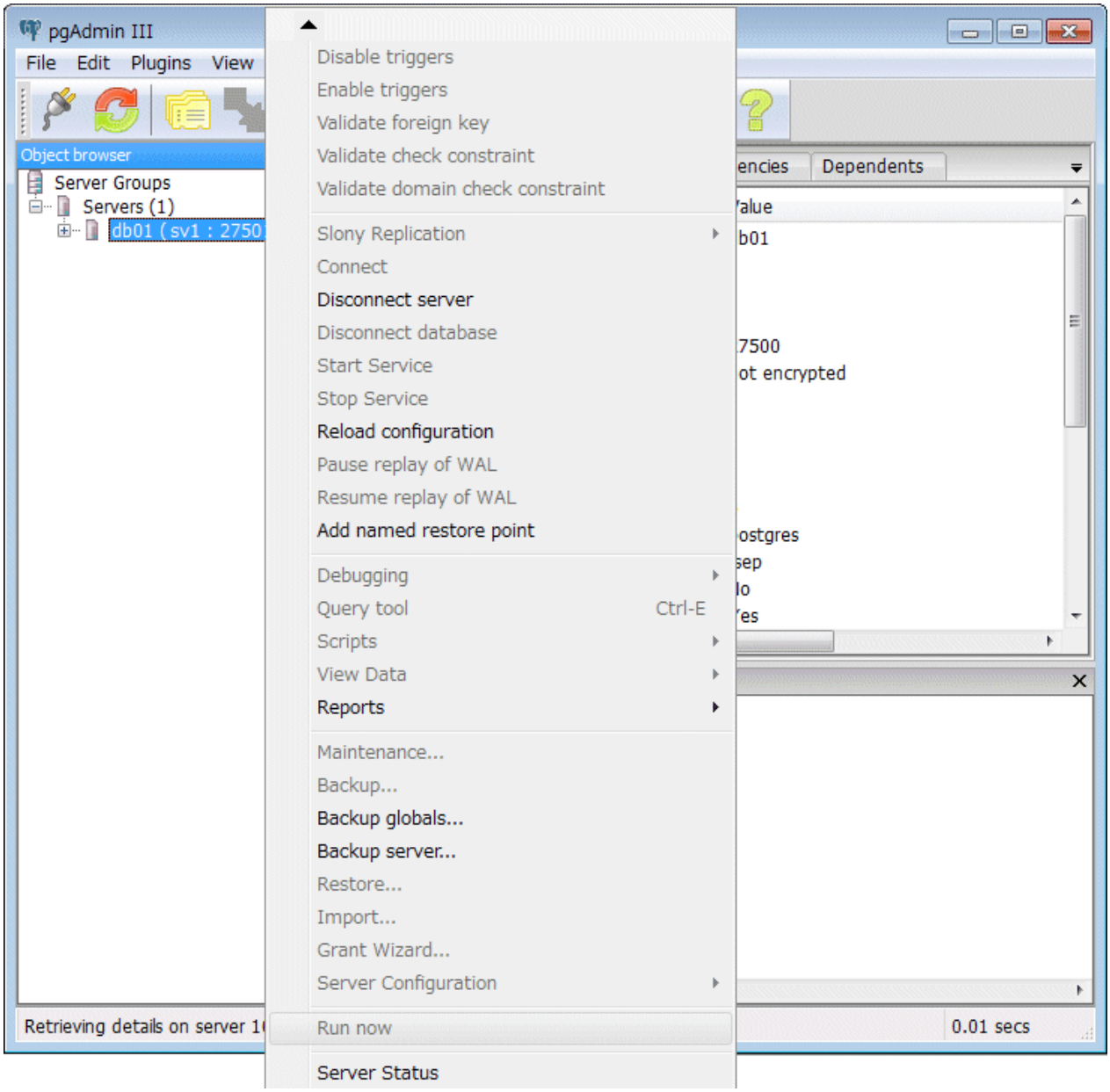
See

- Refer to "Notes on Application Compatibility" in the Application Development Guide for information on maintaining application compatibility.
- Refer to "The Statistics Collector" under "Server Administration" in the PostgreSQL Documentation for information on pg_stat_activity.

6.4.2 Using pgAdmin

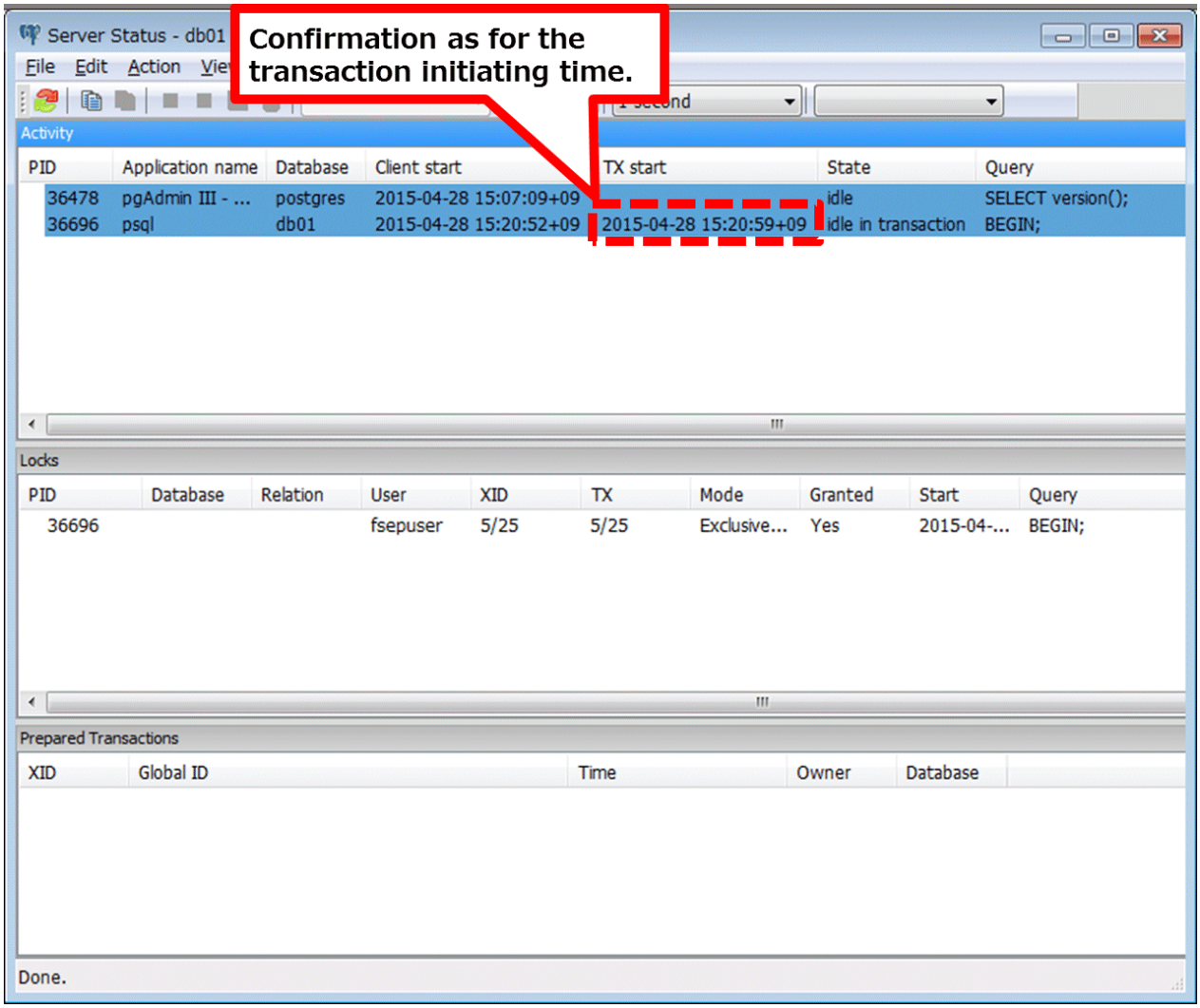
This section describes the procedure for monitoring connections using [Server Status] in pgAdmin.

1. From the [Tools] menu in pgAdmin, click [Server Status].



2. Identify client connections that have been in the waiting state for an extended period.

   From the transaction start time displayed under [TX Start], identify connections that have been in the waiting state for an extended period.



6.5 Reorganizing Indexes

Normally, a database defines indexes in tables, but if data is frequently updated, indexes can no longer use free space in the disk efficiently. This situation can also cause a gradual decline in database access performance.

To rearrange used space on the disk and prevent the database access performance from declining, it is recommended that you periodically execute the REINDEX command to reorganize indexes.

Check the disk usage of the data storage destination using the method described in "6.2 Monitoring Disk Usage and Securing Free Space".

See

Refer to "Routine Reindexing" under "Server Administration" in the PostgreSQL Documentation for information on reorganizing indexes by periodically executing the REINDEX command.

Point

Typically, reorganize indexes once a month at a suitable time such as when conducting database maintenance. Use SQL statements to check index usage. If this usage is increasing on a daily basis, adjust the frequency of recreating the index as compared to the free disk space.

The following example shows the SQL statements and the output.

However, when considering continued compatibility of applications, do not reference system catalogs and functions directly in the following SQL statements. Refer to "Notes on Application Compatibility" in the Application Development Guide for details.

[SQL statements]

```
SELECT
  nspname AS schema_name,
  relname AS index_name,
  round(100 * pg_relation_size(indexrelid) / pg_relation_size(indrelid)) / 100 AS index_ratio,
  pg_size_pretty(pg_relation_size(indexrelid)) AS index_size,
  pg_size_pretty(pg_relation_size(indrelid)) AS table_size
FROM pg_index I
  LEFT JOIN pg_class C ON (C.oid = I.indexrelid)
  LEFT JOIN pg_namespace N ON (N.oid = C.relnamespace)
WHERE
  C.relkind = 'i' AND
  pg_relation_size(indrelid) > 0
ORDER BY pg_relation_size(indexrelid) DESC, index_ratio DESC;
```

[Output]

```
 schema_name |            index_name            | index_ratio | index_size | table_size
-------------+----------------------------------+-------------+------------+------------
 public      | pgbench_accounts_pkey            |        0.16 | 2208 KB    | 13 MB
 pg_catalog  | pg_depend_depender_index         |         0.6 | 224 KB     | 368 KB
 pg_catalog  | pg_depend_reference_index        |        0.58 | 216 KB     | 368 KB
...
```

See

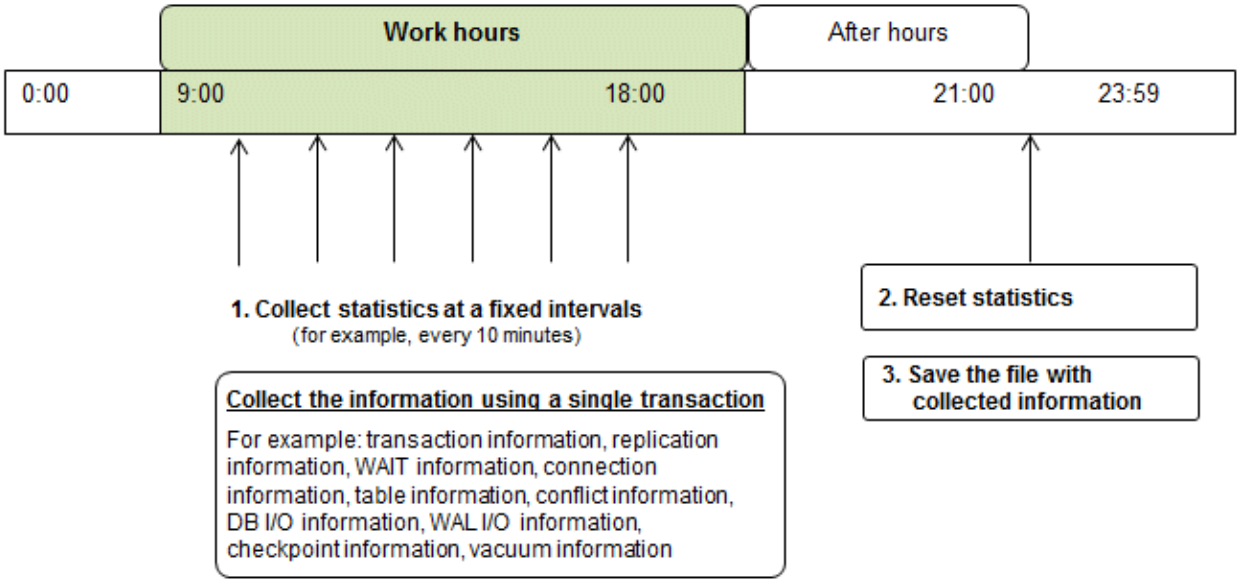
Refer to "Notes on Application Compatibility" in the Application Development Guide for information on maintaining application compatibility.

6.6 Monitoring Database Activity

Enterprise Postgres enables you to collect information related to database activity. By monitoring this information, you can check changes in the database status.

This information includes wait information for resources such as internal locks, and is useful for detecting performance bottlenecks. Furthermore, you should collect this information in case you need to request Fujitsu technical support for an investigation.

Figure 6.1 Overview of information collection



1. Collect statistics at fixed intervals during work hours.

   Accumulate the collected information into a file.

   Wherever possible, collect data from the various statistics views using a single transaction, because it enables you to take a snapshot of system performance at a given moment.

   Refer to "6.6.1 Information that can be Collected" for information on the system views that can be collected.

2. Reset statistics after work hours, that is, after jobs have finished.

   Refer to "6.6.3 Information Reset" for information on how to reset statistics.

3. Save the file with collected information.

   Keep the file with collected information for at least two days, in order to check daily changes in performance and to ensure that the information is not deleted until you have sent a query to Fujitsu technical support.

Where jobs run 24 hours a day, reset statistics and save the file with collected information when the workload is low, for example, at night.

Note

Statistics cumulatively add the daily database value, so if you do not reset them, the values will exceed the upper limit, and therefore will not provide accurate information.

The subsections below explain the following:

- Information that can be collected
- Collection configuration
- Information reset

6.6.1 Information that can be Collected

Information that can be collected is categorized into the following two types:

- Information common to PostgreSQL
- Information added by Enterprise Postgres

Information common to PostgreSQL

See

Refer to "Monitoring Database Activity" under "Server Administration" in the PostgreSQL Documentation for information on information common to PostgreSQL.

Information added by Enterprise Postgres

You can collect the following information added by Enterprise Postgres.

Table 6.1 Information added by Enterprise Postgres

View name	Description
pgx_stat_lwlock	Displays statistic related to lightweight lock, with each type of content displayed on a separate line. This information helps to detect bottlenecks. Refer to "C.2 pgx_stat_lwlock" for details.
pgx_stat_latch	Displays statistics related latches, with each type of wait information within Enterprise Postgres displayed on a separate line. This information helps to detect bottlenecks. Refer to "C.3 pgx_stat_latch" for details.
pgx_stat_walwriter	Displays statistics related to WAL writing, in a single line. Refer to "C.4 pgx_stat_walwriter" for details.
pgx_stat_sql	Displays statistics related to SQL statement executions, with each type of SQL statement displayed on a separate line. Refer to "C.5 pgx_stat_sql" for details.

6.6.2 Collection Configuration

The procedure for configuring collection depends on the information content.

- Information common to PostgreSQL
- Information added by Enterprise Postgres

Information common to PostgreSQL

See

Refer to "The Statistics Collector" in "Monitoring Database Activity" under "Server Administration" in the PostgreSQL Documentation for information on information common to PostgreSQL.

Information added by Enterprise Postgres

Information added by Enterprise Postgres is collected by default.

To enable or disable information collection, change the configuration parameters in postgresql.conf. The following table lists the views for which you can enable or disable information collection, and the configuration parameters.

View name	Parameter
pgx_stat_lwlock pgx_stat_latch	track_waits
pgx_stat_sql	track_sql

Remarks: You cannot change the collection status for pgx_stat_walwriter.

Refer to "Appendix A Parameters" for information on the parameters.

6.6.3 Information Reset

This section describes how to reset information.

Information added by Enterprise Postgres

You can reset information added by Enterprise Postgres by using the pg_stat_reset_shared function in the same way as for information common to PostgreSQL.

Configure the following parameters in the pg_stat_reset_shared function:

Function	Type of return value	Description
pg_stat_reset_shared(text)	void	Reset some cluster-wide statistics counters to zero, depending on the argument (requires superuser privileges). Calling pg_stat_reset_shared('lwlock') will zero all counters shown in pgx_stat_lwlock. Similarly, in the following cases, all values of the pertinent statistics counter are reset: - If pg_stat_reset_shared('latch') is called: All values displayed in pgx_stat_latch - If pg_stat_reset_shared('walwriter') is called: All values displayed in pgx_stat_walwriter - If pg_stat_reset_shared('sql') is called: All values displayed in pgx_stat_sql

See

Refer to "Statistics Functions" in "Monitoring Database Activity" under "Server Administration" in the PostgreSQL Documentation for information on other parameters of the pg_stat_reset_shared function.

Chapter 7 Setting up and Operating PL/extJava

This chapter provides an overview of PL/extJava, and explains how to set up and operate PL/extJava.

7.1 Overview of PL/extJava

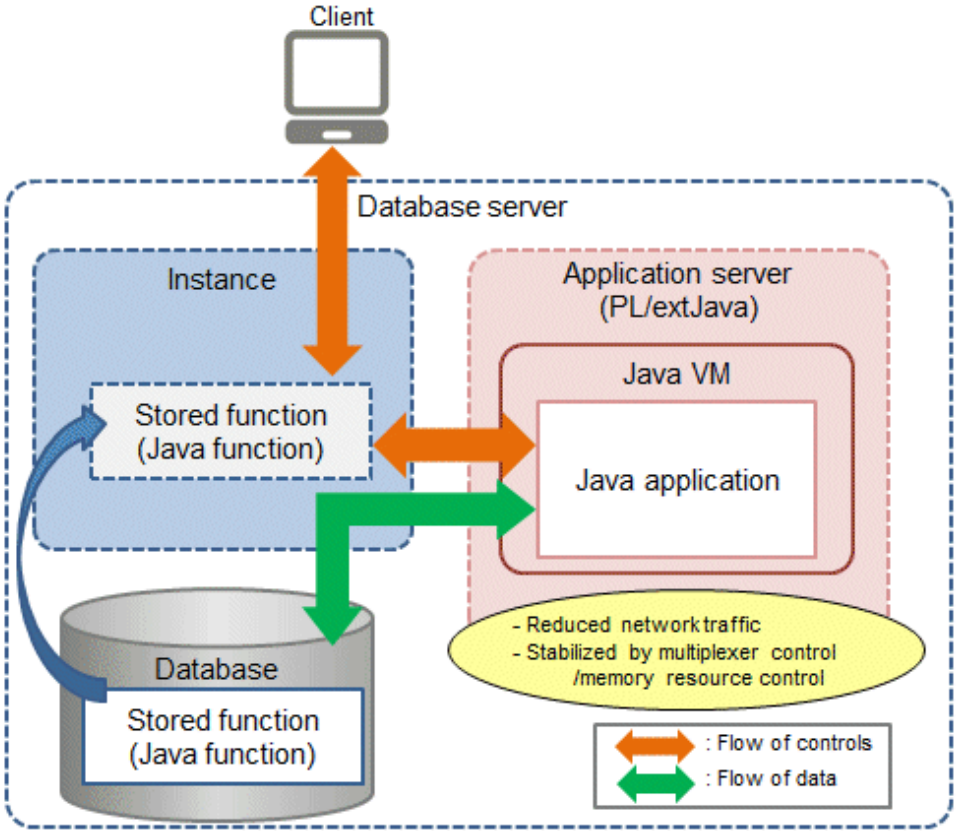
PL/extJava is a framework for incorporating the application server into the database server and controlling the Java Virtual Machine (Java VM).

Jobs where a client frequently accesses a database using SQL mean a high volume of network traffic between them. As a result, not only these jobs may experience a drop in response, but other job systems may as well. It is possible to reduce network traffic and improve the job processing time by performing business logic, implemented via stored functions, on the database server instead of on the client.

However, if several clients access the database server simultaneously, Java VM multiplexer control and memory resources must be taken into account on the database server if Java VM is run.

PL/extJava can control the Java VM multiplexer control and memory resources using the application server, so stored functions can be executed efficiently even in operation modes that have simultaneous access from many clients.

Figure 7.1 Overview of PL/extJava



Point

In this chapter, PL/extJava stored functions that are registered in the database are referred to as "Java functions". Applications that operate on the Java VM are referred to as "Java applications".

Note

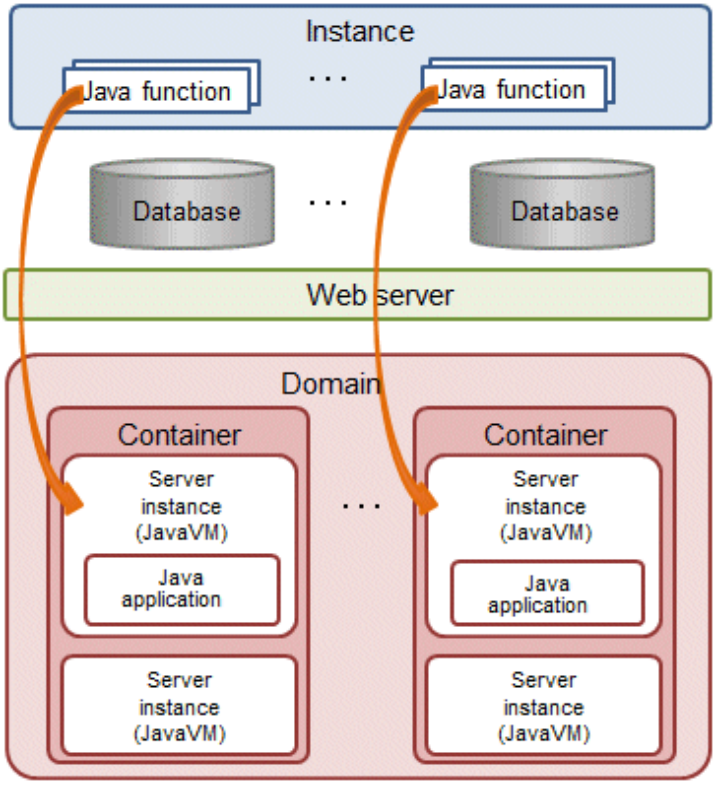
- Java functions run on different connections from the ones between the client and the database server. Jobs being migrated to Java functions must therefore be run as independent transactions.

- Names of databases that use PL/extJava must be specified in up to 28 bytes. Names of roles for connecting from the application server to the database must be specified in up to 8 bytes.

7.1.1 PL/extJava Configuration

This section explains the configuration of PL/extJava.

Figure 7.2 PL/extJava configuration



The elements in the figure are explained below.

- Instance

  An instance of Enterprise Postgres.

  If a Java function is called from the client, the web server is requested to execute the Java application.

  Refer to "7.2.4 Registering Java Functions" for details.

- Web server

  The web server receives the request from the Java function to execute the Java application.

  Based on this request, a server instance (Java VM) in the container is requested to execute the Java application.

  If there are multiple server instances (Java VM) in the container, the request will be allocated to the optimal server.

- Container

  The container is the execution environment for the Java application.

  Each container corresponds to one database of an instance that executes a Java function.

- Server instance (Java VM)

  A server instance (Java VM) is a single Java VM. Java applications can be executed simultaneously on a server instance (Java VM).

  The maximum number of Java applications that can be executed simultaneously can be extended by adding server instances (Java VM).

- Domain

  The domain centrally manages the containers.

  Each domain corresponds to a single instance. The directory that stores the domain resources is known as the "domain root".

7.1.2 Application Servers

An application server based on Interstage Application Server technology, which is incorporated in Enterprise Postgres, can be used to achieve the benefits listed below.

(Hereafter, an application server based on Interstage Application Server technology, which is incorporated in Enterprise Postgres, is referred to as a "Enterprise Postgres Java application server").

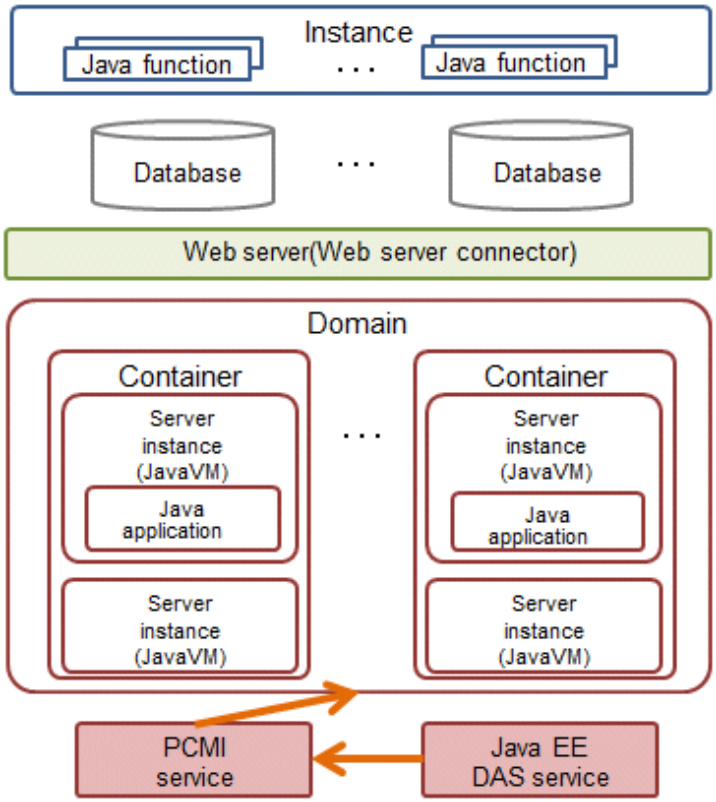
- The pgx_jadmin command can be used. This command simplifies the settings and controls for the Enterprise Postgres Java application server, enabling the user to easily perform setup and tuning, even without a detailed understanding of Enterprise Postgres Java application server.
- Features of Enterprise Postgres Java application server such as those listed below can be used transparently to easily achieve stable operation.
  - If several Java functions are executed, Java applications will be executed on the optimized number of Java VMs, so operations can be performed with the minimum required memory resources.
  - In anticipation of Java full garbage collection, the relevant Java VM can be disconnected to prevent any unexpected delay of Java applications.
  - Java VM heap usage and garbage collection frequency can be monitored, and risk indicators can be reported in alert notifications. (Predictive monitoring feature)
  - Java VM heartbeat monitoring enables the logging of abnormal operations of Java applications, which facilitates the efficient investigation of error causes. (Timer feature)

See

Refer to "pgx_jadmin" in the Reference for information on the pgx_jadmin command.

If using a Enterprise Postgres Java application server, the framework will be configured as shown below.

Figure 7.3 Configuration when using Enterprise Postgres Java application server



The specific elements of Enterprise Postgres Java application server shown in the figure are explained below.

- Web server connector

  The role of the web server connector is to transfer requests received by the web server to the container.

- PCMI service

  This service manages (starts, stops, and monitors) the Java EE DAS service and server instances (Java VM).

- Java EE DAS service

  This service integrates with the PCMI service to manage container operations.

  It receives operation requests from the pgx_jadmin command.

Note

Note the following points when using Enterprise Postgres Java application server:

- Only one domain can be created per machine. Therefore only one database cluster that uses PL/extJava can be created per machine.
- Up to 64 Java applications can be executed simultaneously by a single server instance (Java VM). To simultaneously execute more applications than this, use the pgx_jadmin command to add server instances (Java VM) as required.

7.1.3 User Definitions

This section explains the users required by PL/extJava.

- Instance administrator

  The instance administrator becomes the domain administrator. Furthermore, the instance administrator also becomes the effective user of processes that configure the application server container, so that operations (such as file access via Java applications) are performed with instance administrator privileges.

  If using the features of Enterprise Postgres Java application server incorporated into Enterprise Postgres:

  - This is the effective user of containers and the pgx_jadmin command (excluding the init-domain, delete-domain, backup and restore subcommands).
  - Specify this user in the "--dbadminuser" option of "pgx_jadmin init-domain".
- Users who connect to the database

  If Java applications access the database, use the JDBC connection pool. These are the authenticated users for this access.

  Use the CREATE USER statement to add users who connect to the database.

  Refer to "7.2.3.1 Configuring Database Clusters" if using a Enterprise Postgres Java application server.

7.2 Setting up PL/extJava

This section explains the setting up of PL/extJava.

Perform the procedure below to set up PL/extJava if using Enterprise Postgres Java application server.

Note

- PL/extJava cannot be set up using WebAdmin or pgAdmin.
- Only operations in the IPv6/IPv4 dual stack environment are supported. Operations are not supported if IPv4 is disabled.

Setup flow

Perform the following procedure to set up PL/extJava.

Create a database cluster in advance.

1. Preparing Port Numbers
2. Creating Domains
3. Creating PL/extJava
4. Registering Java Functions

7.2.1 Preparing Port Numbers

This section explains the port numbers used in PL/extJava.

Port numbers for domain management

Usage	Default port number	Specification method	Description
Port number for domain management	27530	pgx_jadmin init-domain or pgx_jadmin modify-domain-port	Port numbers for domain management. Three port numbers are required.
	27521		
	27522		

See

Refer to "7.2.2 Creating Domains" for information on pgx_jadmin init-domain.

Refer to "7.3.3.3 Changing Port Numbers" for information on pgx_jadmin modify-domain-port.

Port numbers for server instance (Java VM) management

The following port numbers are required for each server instance (Java VM).

Usage	Default port number	Specification method	Description
Port number for server instance (Java VM) management	27531 27532	pgx_jadmin create-container --instanceport, pgx_jadmin add-instance --instanceport, or pgx_jadmin modify-instance-port --instanceport	Used for operating a server instance (Java VM). Two port numbers must be configured for each server instance (Java VM).

See

Refer to "7.2.3.2 Creating Containers" for information on pgx_jadmin create-container.

Refer to "7.3.3.1 Adding or Deleting Server Instances (Java VM)" for information on pgx_jadmin add-instance.

Refer to "7.3.3.3 Changing Port Numbers" for information on pgx_jadmin modify-domain-port.

Note

- For the port number, specify an unused port number in the following range:
  - 1024 to 32767
- Configuring the firewall

  The port number specified in the pgx_jadmin command is used for communication within the local machine. Use the firewall to restrict access to each port number. This prevents unauthorized access and ensures security.

7.2.2 Creating Domains

Create a domain for using PL/extJava.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Edit the hosts file

   Add the information of the host that builds PL/extJava to the hosts file.

3. Create the domain

   Use the init-domain subcommand of the pgx_jadmin command to create the domain.

   ### Example

   If the domain root is "/domain" and the instance administrator name is "fsepuser "

   ```
   # pgx_jadmin init-domain --shareddir /domain --dbadminuser fsepuser --domainport
   27530,27521,27522 --pgdata /database/inst1
   ```

See

Refer to "pgx_jadmin" in the Reference for information on the init-domain subcommand.

7.2.3 Creating PL/extJava

This section explains how to create PL/extJava.

7.2.3.1 Configuring Database Clusters

Configure the definition of the database cluster.

1. Configure the "listen_addresses" setting in the postgresql.conf file

   A loopback address is used to connect the Java application to the database. Therefore, ensure that "listen_addresses" is configured to allow connection from localhost.

   See

   Refer to "Connections and Authentication" under "Server Administration" in the PostgreSQL Documentation for details.

2. Add users who connect to the database

   If Java applications access the database, use the JDBC connection pool.

   Use the CREATE USER statement to add users who connect to the database. Specify the additional user names when creating the container.

   Example

   If the user name is "user01"

   ```
   db01=# create user user01;
   ```

3. Configure the pg_hba.conf file

   Establish a local connection to the database from the Java application. Ensure that the settings allow local connections.

Parameter	Parameter value
TYPE	"host"
DATABASE	Name of the database the Java application connects to
USER	User name used when the Java application is connected to the database
ADDRESS	127.0.0.1/32
AUTH-METHOD	Use a method other than trust authentication

   See

   Refer to "The pg_hba.conf File" under "Server Administration" in the PostgreSQL Documentation for details.

4. Configure "max_connections" in the postgresql.conf file

   Add the number of connections used when connecting from Enterprise Postgres Java application server to the database, calculated as shown below, to the existing value of max_connections.

   ```
   numberOfJavaVmServerInstances x 64
   ```

See

Refer to "Configuring Remote Connections" in the Installation and Setup Guide for Server for details.

7.2.3.2 Creating Containers

Create one container per database within a database cluster that uses Java functions.

When you create a container, a server instance (Java VM) is created in the container at the same time. Settings for the JDBC connection pool and the data source used to access the database from the Java application are also configured at this time. The name of the server instance (Java VM) will be "*databaseName-serialNumber*". Confirm the name of the server instance (Java VM) using the list-container subcommand of the pgx_jadmin command.

See

Refer to "pgx_jadmin" in the Reference for information on the list-container subcommand.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Create the container

   Use the create-container subcommand of the pgx_jadmin command to create the container. Information required for the database connection can still be modified later.

   Example

   ```
   $ pgx_jadmin create-container --dbname db01 --dbuser user01 --dbpassword password1 --
   instanceport 27531,27532
   ```

   See

   Refer to "pgx_jadmin" in the Reference for information on the create-container subcommand.

3. Add the server instances (Java VM)

   To simultaneously execute 65 or more Java applications, add the required extra number of server instances (Java VM).

   Refer to "7.3.3.1 Adding or Deleting Server Instances (Java VM)" for information on adding server instances (Java VM).

4. Start the instance again

   Refer to "2.1 Starting and Stopping an Instance" for information on how to start the instance.

   Refer to "8.10 Actions in Response to Instance Startup Failure" if the instance fails to start.

5. Install the PL/extJava extension

   Execute the CREATE EXTENSION statement and install the PL/extJava extension.

   ```
   db01=# CREATE EXTENSION plextjavau;
   ```

7.2.4 Registering Java Functions

This section explains how to register Java functions and store Java applications.

This procedure must be performed each time a Java function is created.

Registering Java Functions

1. Connect to the database

   As the instance administrator, connect to the database that corresponds to the container that was created using the psql command or pgAdmin.

2. Register the function in the database

   Java functions run on sessions that are not dependent on the caller session. To operate a Java function using the caller session, use the SET statement to set the plextjava.separate_session parameter to "off" and then register it.

```
SET plextjava.separate_session=off
```

   This section explains the CREATE FUNCTION statement used when registering a function in the database.

   Syntax:

```
CREATE  [ OR  REPLACE ]  FUNCTION
    name  ( [ IN ] argtype [, ...] )
    [ RETURNS  rettype ]
    AS  'definition'
    LANGUAGE  lang_name
```

   name

   Specify the name of the Java function.

   The name of the Java function does not need to match the name of the Java application method.

   A qualified schema name can be used. Create the schema in advance.

   argtype

   Specify the data type of the Java function argument.

   rettype

   Specify the data type of the Java function return value.

   definition

   Specify "*packageName.className. methodName*" or "*className.methodName"* of the Java application.

   Do not specify the method arguments as they will be invalid.

   lang_name

   Specify "plextjavau".

Example

If the schema name is "javatest", the Java function name is "java_addOne", and the "*packageName.className.methodName*" of the Java application is "org.postgresql.plextjava.example.Parameters.addOne"

```
db01=# CREATE FUNCTION javatest.java_addOne(INTEGER)
                  RETURNS INTEGER
                  AS 'org.postgresql.plextjava.example.Parameters.addOne'
                  LANGUAGE plextjavau;
```

See

- Refer to "Relationship between the Java Function Data Type and Application Data Type" in the Application Development Guide for information on data types.
- Refer to "CREATE FUNCTION" under "Reference" in the PostgreSQL Documentation for details.

3. Set privileges for users connecting to the database

   Grant privileges to users for database objects (such as tables, columns, and views) that will be accessed when accessing the database from the Java application. Use the GRANT statement to grant privileges.

   Example

   Grant privileges to user name "user01" to add data to table "table01"

   ```
   db01=# GRANT INSERT ON table01  to user01;
   ```

4. Set privileges for Java functions

   Grant execution privileges for Java functions to users who call Java functions. Use the GRANT statement to grant privileges.

   Note

   Grant execution privileges for a Java function only to users who have permission to access the resources accessed by that functions, otherwise the users may inadvertently be granted improper access to those resources via the function.

   Example

   Grant privileges to user "user04" for the Java function "java_addOne"

   ```
   db01=# GRANT EXECUTE ON FUNCTION javatest.java_addOne(INTEGER) to user04;
   ```

5. Disconnect from the database.

See

Refer to "Reference" in the PostgreSQL Documentation for information on the GRANT statement.

Storing Java applications

1. Copy the Java application

   Manually copy the Java application in jar format to the directory below.

   ```
   domainRoot/plextjava/databaseName
   ```

7.3 PL/extJava Operation

This section explains the operation of PL/extJava.

If using the Enterprise Postgres Java application server, operate using the procedure explained below.

7.3.1 Starting and Stopping Containers

The container will start or stop automatically when the instance is started or stopped.

7.3.2 Checking PL/extJava

This section explains how to check PL/extJava.

7.3.2.1 Checking the Domain Information

Use the list-domain subcommand of the pgx_jadmin command to check the domain information.

 Example

```
$ pgx_jadmin list-domain
```

Sample display

```
domain status: running
shareddir: /domain
domainport: 27530,27521,27522
dbadminuser: fsepuser
datadir: /database/inst1
```

 See

Refer to "pgx_jadmin" in the Reference for information on the list-domain subcommand.

7.3.2.2 Checking the Container Information

Use the list-container subcommand of the pgx_jadmin command to check the container information.

 Example

If the database name is "db01"

```
$ pgx_jadmin list-container --dbname db01
```

Sample display

```
container status: running
dbname: db01
instance: db01-1 27531,27532
instance: db01-2 27801,27802
dbport: 27011
dbuser: user01
```

 See

Refer to "pgx_jadmin" in the Reference for information on the list-container subcommand.

7.3.3 Changing PL/extJava

This section explains how to change PL/extJava.

7.3.3.1 Adding or Deleting Server Instances (Java VM)

This section explains how to add or delete a server instance (Java VM).

A server instance (Java VM) is a single Java VM. Up to 64 Java applications can be executed simultaneously on a single server instance (Java VM).

The number of Java applications that can be executed simultaneously can be extended by adding server instances (Java VM). To decrease the number of Java applications that are executed simultaneously, delete the server instances (Java VM) that were added.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Change the database cluster configuration

   Change the maximum number of simultaneous instance connections (max_connections).

   Refer to "7.2.3.1 Configuring Database Clusters" for information on how to change the number of simultaneous connections.

3. Add or delete the server instance (Java VM)

   Add or delete the server instance (Java VM) using the add-instance subcommand or the delete-instance subcommand of the pgx_jadmin command.

   Example

   Add a server instance (Java VM)

   ```
   $ pgx_jadmin add-instance --dbname db01 --instanceport 27801,27802
   ```

   Delete a server instance (Java VM)

   ```
   $ pgx_jadmin delete-instance --dbname db01
   ```

   See

   Refer to "pgx_jadmin" in the Reference for information on the add-instance and delete-instance subcommands.

4. Start the instance again

   Refer to "2.1 Starting and Stopping an Instance" for information on how to start the instance.

   Refer to "8.10 Actions in Response to Instance Startup Failure" if the instance fails to start.

7.3.3.2 Changing the Database Connection Information

This section explains how to change database connection information.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Change the database connection information

   Use the modify-container-db subcommand of the pgx_jadmin command to change the database connection information.

   You can change the following database connection information:

   - Instance port number
   - User name used for database connection
   - User password used for database connection

   Example

   If changing the name of the user connecting to the database to "user02"

   ```
   $ pgx_jadmin modify-container-db --dbname db01 --dbuser user02
   ```

See

Refer to "pgx_jadmin" in the Reference for information on the modify-container-db subcommand.

3. Start the instance again

   After changing the instance connection information, start the instance.

   Refer to "2.1 Starting and Stopping an Instance" for information on how to start the instance.

   Refer to "8.10 Actions in Response to Instance Startup Failure" if the instance fails to start.

7.3.3.3 Changing Port Numbers

This section explains how to change the domain management port numbers, or the port numbers for managing a server instance (Java VM).

Changing the domain management port numbers

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Check the port numbers before they are changed

   Use the list-domain subcommand of the pgx_jadmin command to check the port numbers before they are changed.

   Example

```
$ pgx_jadmin list-domain
```

   See

   Refer to "pgx_jadmin" in the Reference for information on the list-domain subcommand.

3. Change the domain management port numbers

   Use the modify-domain-port subcommand of the pgx_jadmin command to change the port numbers.

   Example

   If changing the domain management port numbers to "26600,26601,26602"

```
$ pgx_jadmin modify-domain-port --domainport 27600,27601,27602
```

   See

   Refer to "pgx_jadmin" in the Reference for information on the modify-domain-port subcommand.

4. Check the port numbers after changing them

   Use the list-domain subcommand of the pgx_jadmin command to check that the port numbers were changed.

5. Start the instance again

   Refer to "2.1 Starting and Stopping an Instance" for information on how to start the instance.

Refer to "8.10 Actions in Response to Instance Startup Failure" if the instance fails to start.

Changing the server instance (Java VM) management port numbers

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Check the port numbers before they are changed

   Use the list-container subcommand of the pgx_jadmin command to check the port numbers before they are changed.

    Example

   ```
   $ pgx_jadmin list-container --dbname db1
   ```

    See

   Refer to "pgx_jadmin" in the Reference for information on the list-container subcommand.

3. Change the server instance (Java VM) management port numbers

   Use the modify-instance-port subcommand of the pgx_jadmin command to change the port numbers.

    Example

   To change the server instance (Java VM) management port numbers to "26701,26702"

   ```
   $ pgx_jadmin modify-instance-port --instance db1-1 --instanceport 27701,27702
   ```

    See

   Refer to "pgx_jadmin" in the Reference for information on the modify-instance-port subcommand.

4. Check the port numbers after changing them

   Use the list-container subcommand of the pgx_jadmin command to check that the port numbers were changed.

5. Start the instance again

   Refer to "2.1 Starting and Stopping an Instance" for information on how to start the instance.

   Refer to "8.10 Actions in Response to Instance Startup Failure" if the instance fails to start.

7.3.4 Deleting Java Functions

This section explains how to delete Java functions and Java applications from PL/extJava.

 Point

Java functions can be deleted while the instance is running.

Delete a Java function

1. Connect to the database

   As the instance administrator, connect to the database that corresponds to the container that was created using the psql command or pgAdmin.

2. Delete the Java function

   Delete the Java function that was registered in the database.

```
db01=# DROP FUNCTION javaFunctionName(argument);
```

3. Disconnect from the database.

Refer to "DROP FUNCTION" under "Reference" in the PostgreSQL Documentation for details.

Delete the Java application

1. Delete the Java application
   Delete the jar file of the following directory.

```
domainRoot/plextjava/databaseName
```

7.3.5 Deleting Containers

This section explains how to delete containers.

1. Delete the Java function

   As the instance administrator, connect to the database that corresponds to the container that was created.

   Delete all Java functions that were registered in the database.

   Refer to "7.3.4 Deleting Java Functions" for details.

2. Uninstall the PL/extJava extension

   As the instance administrator, connect to the database that corresponds to the container that was created.

   Execute the DROP EXTENSION statement and uninstall the PL/extJava extension.

```
db01=# DROP EXTENSION plextjavau;
```

3. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

4. Delete the container

   Use the delete-container subcommand of the pgx_jadmin command to delete the container.

```
$ pgx_jadmin delete-container --dbname db01
```

Refer to "pgx_jadmin" in the Reference for information on the delete-container subcommand.

7.3.6 Deleting Domains

This section explains how to delete domains.

Note

When a domain is deleted, the domain root directory is also deleted. If any files in this directory are required, such as the Java applications, perform a backup prior to deleting the domain.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Delete the domain

   Use the delete-domain subcommand of the pgx_jadmin command to delete the domain.

   ### Example

   ```
   # pgx_jadmin delete-domain
   ```

   ### See

   Refer to "pgx_jadmin" in the Reference for information on the delete-domain subcommand.

7.3.7 Backup and Restore

This section explains how to back up and restore PL/extJava,

Back up PL/extJava to guard against issues such as a failure of the disk where the PL/extJava domains and containers are placed.

For example, backup should be performed when any of the following occur:

- Modifications are made to PL/extJava (such as to its containers or server instances (Java VM))
- Java functions are added, changed, or deleted

Note

When re-creating the instance administrator after a system disk failure, ensure that it has the same user name, UID, and GID as before the failure. If these values are different from the backup data, restoration will fail.

7.3.7.1 Backup Method

This section explains how to back up PL/extJava.

If using the failover operation, back up the active node.

If using database multiplexing mode, back up the primary server.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Prepare the backup storage disk for PL/extJava

   Prepare a disk separate from the disk used to store the PL/extJava resources and mount it using a command such as the OS command. Ensure that the disk has at least twice the capacity of the domain root.

3. Back up PL/extJava

   As the system administrator, back up the PL/extJava resources using the backup subcommand of the pgx_jadmin command.

Note

Do not specify any of the following directories as the PL/extJava backup storage directory:

- The instance data storage destination, the backup storage destination, or the domain root
- A directory in the instance data storage destination, the backup storage destination, or the domain root
- A directory that places the instance data storage destination, the backup storage destination, and the domain root in a subdirectory of a directory that stores the PL/extJava backup

Example

If the backup storage directory for PL/extJava is "/backup"

```
# pgx_jadmin backup --backupdir /backup
```

See

Refer to "pgx_jadmin" in the Reference for information on the backup subcommand.

7.3.7.2 Restore Method

This section explains how to restore PL/extJava from the point it was backed up.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Restore the instance

   Restore the instance if required.

3. Restore PL/extJava

   As the system administrator, restore the PL/extJava resources using the restore subcommand of the pgx_jadmin command.

Example

If the backup storage directory for PL/extJava is "/backup"

```
# pgx_jadmin restore --backupdir /backup
```

See

Refer to "pgx_jadmin" in the Reference for information on the restore subcommand.

Chapter 8 Actions when an Error Occurs

This chapter describes the actions to take when an error occurs in the database or an application, while Enterprise Postgres is operating.

Depending on the type of error, it may be necessary to recover the database cluster. The recovery process recovers the following resources:

- Data storage destination
- Transaction log storage destination (if the transaction log is stored in a separate disk from the data storage destination)
- Backup data storage destination

Note

Even if a disk is not defective, the same input-output error messages, as those generated when the disk is defective, may be output. The recovery actions differ for these error messages.

Check the status of the disk, and select one of the following actions:

- If the disk is defective

  Refer to "8.1 Recovering from Disk Failure (Hardware)", and take actions accordingly.

- If the disk is not defective

  Refer to "8.13 I/O Errors Other than Disk Failure", and take actions accordingly.

  A few examples of errors generated even if the disk is not defective include:

  - Network error with an external disk
  - Errors caused by power failure or mounting issues

Determining the cause of an error

If an error occurs, refer to the WebAdmin message and the server log, and determine the cause of the error.

See

Refer to "Configuring Parameters" in the Installation and Setup Guide for Server for information on server logs.

Approximate recovery time

The formulas for deriving the approximate recovery time of resources in each directory are given below.

- Data storage destination or transaction log storage destination

```
Recovery time = (usageByTheDataStorageDestination + usageByTheTransactionLogStorageDestination) /
diskWritePerformance x 1.5
```

  - *usageByTheDataStorageDestination*: Disk space used by the database cluster
  - *usageByTheTransactionLogStorageDestination*: Disk space used by the transaction log stored outside the database cluster
  - *diskWritePerformance*: Measured maximum data volume (bytes/second) that can be written per second in the system environment where the operation is performed
  - 1.5: Coefficient assuming the time excluding disk write, which is the most time-consuming step

- Backup data storage destination

```
Recovery time = usageByTheBackupDataStorageDestination / diskWritePerformance x 1.5
```

  - *usageByTheBackupDataStorageDestination*: Disk space used by the backup data

- *diskWritePerformance*: Measured maximum data volume (bytes/second) that can be written per second in the system environment where the operation is performed
- 1.5: Coefficient assuming the time excluding disk write, which is the most time-consuming step

8.1 Recovering from Disk Failure (Hardware)

This section describes how to recover database clusters to a point immediately before failure, if a hardware failure occurs in the data storage disk or the backup data storage disk.

There are two methods of recovery:

- 8.1.1 Using WebAdmin
- 8.1.2 Using Server Command

Point

Back up the database cluster after recovering it. Backup deletes obsolete archive logs (transaction logs copied to the backup data storage destination), freeing up disk space and reducing the recovery time.

8.1.1 Using WebAdmin

Recover the database cluster by following the appropriate recovery procedure below for the disk where the failure occurred.

If failure occurred in the data storage disk or the transaction log storage disk

Follow the procedure below to recover the data storage disk or the transaction log storage disk.

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance. Refer to "2.1.1 Using WebAdmin" for information on how to stop an instance. WebAdmin automatically stops instances if recovery of the database cluster is performed without stopping the instance.

3. Recover the failed disk

   Replace the disk, and then recover the volume configuration information.

4. Create a tablespace directory

   If a tablespace was defined after backup, create a directory for it.

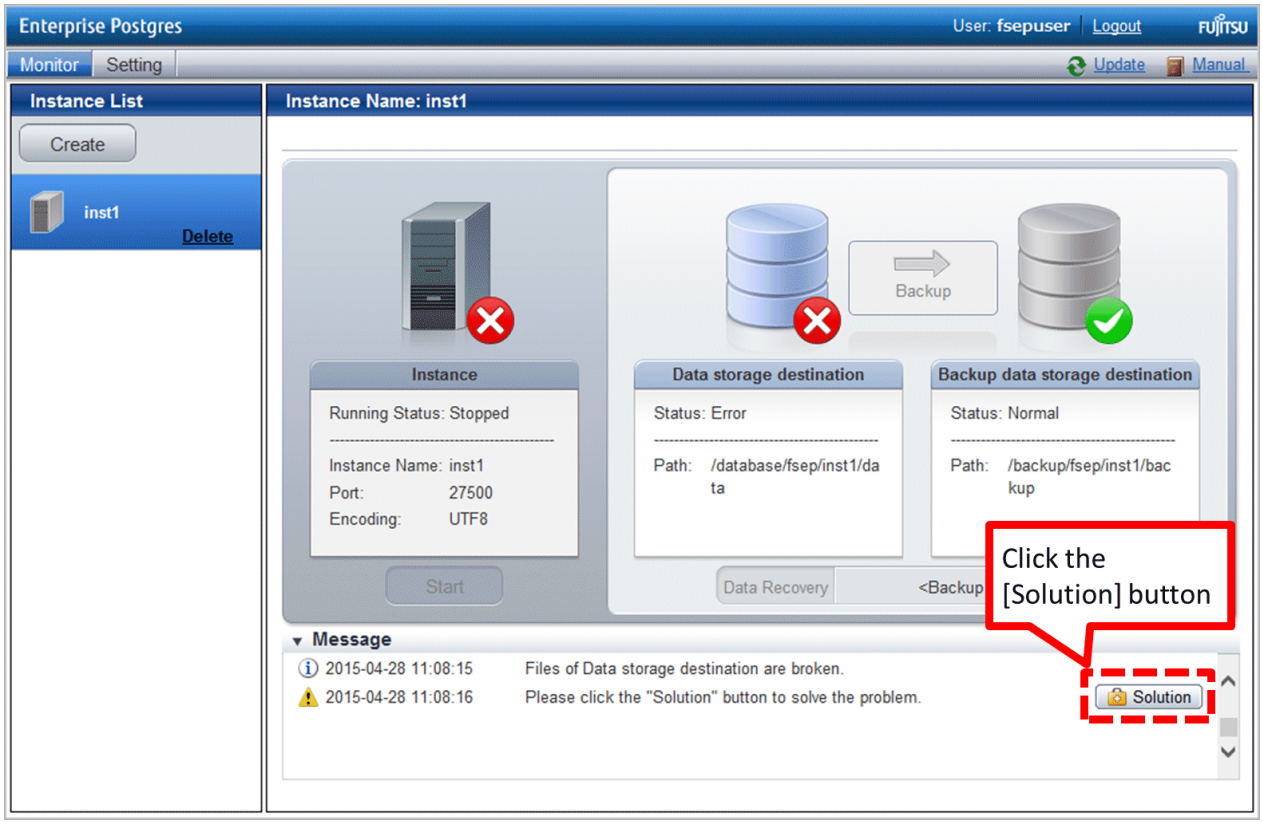
5. Recover the keystore, and enable automatic opening of the keystore

   Do the following if the data in the database has been encrypted:

   - Restore the keystore to its state at the time of the database backup.
   - Enable automatic opening of the keystore.

6. Recover the database cluster

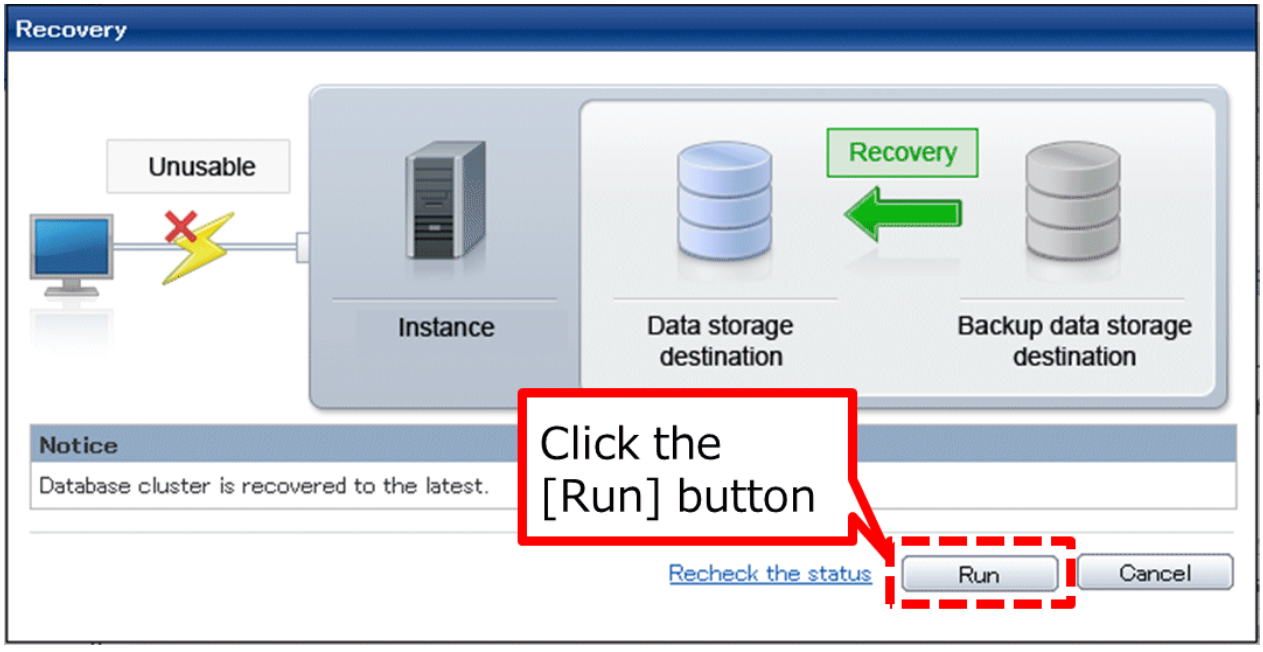
   Log in to WebAdmin, and in the [Monitor] window, click the [Solution] button for the error message.



7. Run recovery

   In the [Recovery] dialog box that appears, click [Run].

   [Recovering] is displayed in the [Monitor] window, and recovery is performed. An instance is automatically started when recovery is successful.



Note

WebAdmin does not support recovery of hash index. If you are using a hash index, then after recovery, execute the REINDEX command to rebuild it. Use of hash indexes is not recommended.

8. Resume applications

   Resume applications that are using the database.

Point

WebAdmin may be unable to detect disk errors, depending on how the error occurred.

If this happens, refer to "8.10.4 Other Errors" to perform recovery.

If failure occurred on the backup data storage disk

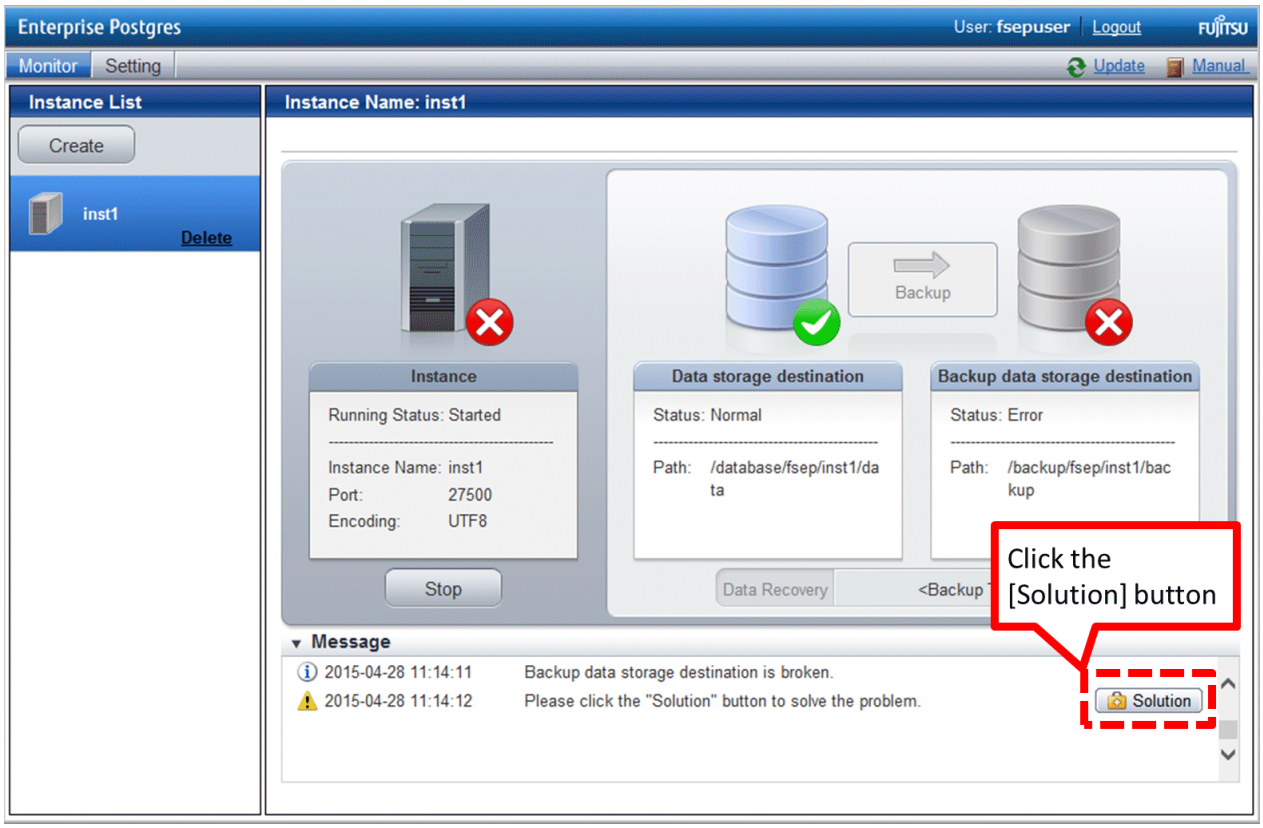
Follow the procedure below to recover the backup data storage disk.

1. Recover the failed disk

   Replace the disk, and then recover the volume configuration information.

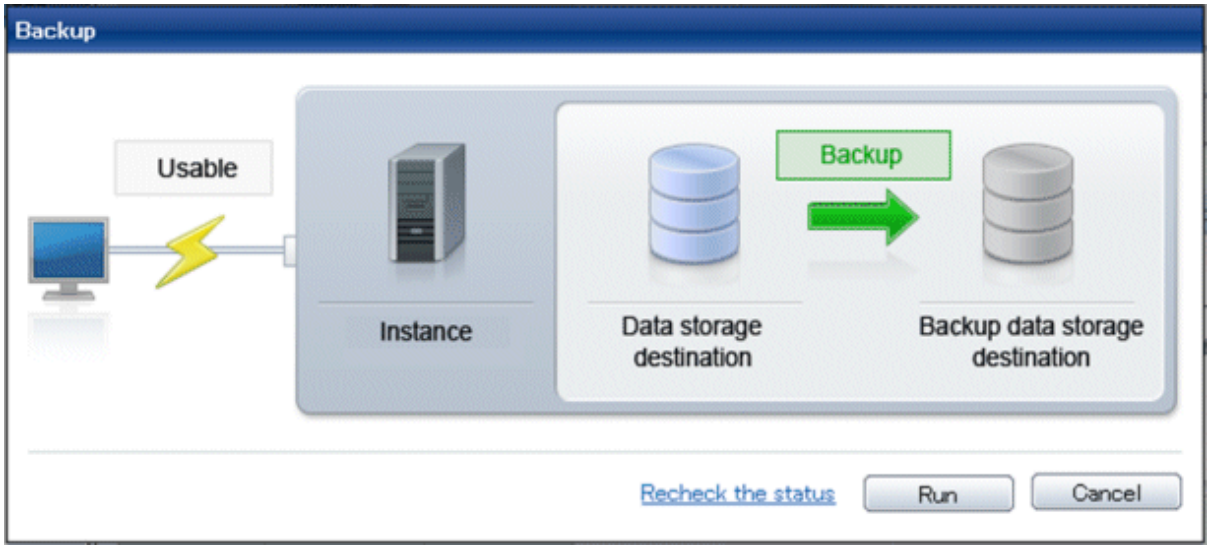
2. Recover the backup data

   Log in to WebAdmin, and in the [Monitor] window, click the [Solution] button for the error message.



3. Run backup

   Perform backup to enable recovery of the backup data. In the [Backup] dialog box that appears, click [Run]. [Backuping] is displayed in the [Monitor] window, and the backup is performed. An instance is automatically started when backup is performed.



Point

If you click [Recheck the status], the resources in the data storage destination and the backup data storage destination are reconfirmed. As a result, the following occurs:

- If an error is not detected

  The status of the data storage destination and the backup data storage destination returns to normal, and it is possible to perform operations as usual.

- If an error is detected

  An error message is displayed in the message list again. Click the [Solution] button, and resolve the problem by following the resolution for the cause of the error displayed in the dialog box.

8.1.2 Using Server Command

Recover the database cluster by following the appropriate recovery procedure below for the disk where the failure occurred.

If failure occurred on the data storage disk or the transaction log storage directory

Follow the procedure below to recover the data storage disk or the transaction log storage directory.

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance, refer to "2.1.2 Using Server Commands" for details.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

3. Recover the failed disk

   Replace the disk, and then recover the volume configuration information.

4. Create a storage destination directory

   - If failure occurred on the data storage disk
     Create a data storage destination directory. If a tablespace was defined, also create a directory for it.

- If failure occurred on the translation log storage disk
  Create a transaction log storage destination directory.

Example

To create a data storage destination directory:

```
$ mkdir /database/inst1
$ chown fsepuser:fsepuser /database/inst1
$ chmod 700 /database/inst1
```

See

Refer to "Preparing Directories to Deploy Resources" under "Setup" in the Installation and Setup Guide for Server for information on how to create a storage directory.

5. Recover the keystore, and enable automatic opening of the keystore

   When the data in the database has been encrypted, restore the keystore to its state at the time of the database backup. Configure automatic opening of the keystore as necessary.

6. Recover the database cluster

   Recover the database cluster using the backup data.

   Specify the following in the pgx_rcvall command:

   - Specify the data storage location in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
   - Specify the backup data storage location in the -B option.

Example

```
> pgx_rcvall -D /database/inst1 -B /backup/inst1
```

Note

If recovery fails, remove the cause of the error in accordance with the displayed error message and then re-execute the pgx_rcvall command.

If the message "pgx_rcvall: an error occurred during recovery" is displayed, then the log recorded when recovery was executed is output after this message. The cause of the error is output in around the last fifteen lines of the log, so remove the cause of the error in accordance with the message and then re-execute the pgx_rcvall command.

The following message displayed during recovery is output as part of normal operation of pgx_rcvall command (therefore the user does not need not be concerned).

```
FATAL:  The database system is starting
```

7. Start the instance

   Start the instance.

   Refer to "2.1.2 Using Server Commands" for information on how to start an instance.

8. Resume applications

   Resume applications that are using the database.

If failure occurred on the backup data storage disk

The procedure for recovering the backup data storage disk is described below.

There are two methods of taking action:

- Performing recovery while the instance is active
- Stopping the instance before performing recovery

The following table shows the different steps to be performed depending on whether you stop the instance.

No	Step	Instance stopped	
		No	Yes
1	Confirm that transaction log mirroring has stopped	Y	N
2	Stop output of archive logs	Y	N
3	Stop applications	N	Y
4	Stop the instance	N	Y
5	Recover the failed disk	Y	Y
6	Create a backup data storage destination directory	Y	Y
7	Resume output of archive logs	Y	N
8	Resume transaction log mirroring	Y	N
9	Start the instance	N	Y
10	Run backup	Y	Y
11	Resume applications	N	Y

Y: Required
N: Not required

The procedure is as follows:

If an instance has not been stopped

1. Confirm that transaction log mirroring has stopped

   Use the following SQL function to confirm that transaction log mirroring has stopped.

```
 postgres=# SELECT pgx_is_wal_multiplexing_paused();
 pgx_is_wal_multiplexing_paused
-------------------
 t
(1 row)
```

   If transaction log mirroring has not stopped, then stop it using the following SQL function.

```
postgres=# SELECT pgx_pause_wal_multiplexing();
LOG:  multiplexing of transaction log files has been stopped
pgx_pause_wal_multiplexing
---------------------------

(1 row)
```

2. Stop output of archive logs

   Transaction logs may accumulate during replacement of backup storage disk, and if the data storage disk or the transaction log storage disk becomes full, there is a risk that operations may not be able to continue.

   To prevent this, use the following methods to stop output of archive logs.

   - Changing archive_command

     Specify a command that will surely complete normally, such as "echo skipped archiving WAL file %f" or "/bin/true", so that archive logs will be regarded as having been output.

If you specify echo, a message is output to the server log, so it may be used as a reference when you conduct investigations.

- Reload the configuration file

  Execute the pg_ctl reload command or the pg_reload_conf SQL function to reload the configuration file.

If you simply want to stop output of errors without the risk that operations will not be able to continue, specify an empty string ('') in archive_command and reload the configuration file.

3. Recover the failed disk

   Replace the disk, and then recover the volume configuration information.

4. Create a backup data storage destination

   Create a backup data storage destination.

   Example

   ```
   $ mkdir /database/inst1
   $ chown fsepuser:fsepuser /database/inst1
   $ chmod 700 /database/inst1
   ```

   Refer to "3.2.2 Using Server Commands" for information on how to create a backup data storage destination.

5. Resume output of archive logs

   Return the archive_command setting to its original value, and reload the configuration file.

6. Resume transaction log mirroring

   Execute the pgx_resume_wal_multiplexing SQL function.

   Example

   ```
   SELECT pgx_resume_wal_multiplexing()
   ```

7. Run backup

   Use the pgx_dmpall command to back up the database cluster.

   Specify the following value in the pgx_dmpall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

   Example

   ```
   > pgx_dmpall -D /database/inst1
   ```

If an instance has been stopped

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance. Refer to "2.1.2 Using Server Commands" for details.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

3. Recover the failed disk

   Replace the disk, and then recover the volume configuration information.

4. Create a backup data storage destination

   Create a backup data storage destination.

Example

```
# mkdir /backup/inst1
# chown fsepuser:fsepuser /backup/inst1
# chmod 700 /backup/inst1
```

Refer to "3.2.2 Using Server Commands" for details.

5. Start the instance

   Start the instance. Refer to "2.1.2 Using Server Commands" for information on how to start an instance.

6. Run backup

   Use the pgx_dmpall command to back up the database cluster.

   Specify the following value in the pgx_dmpall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

   Example

```
> pgx_dmpall -D /database/inst1
```

7. Resume applications

   Resume applications that are using the database.

See

- Refer to "pgx_rcvall" and "pgx_dmpall" in the Reference for information on the pgx_rcvall command and pgx_dmpall command.
- Refer to "Write Ahead Log" under "Server Administration" in the PostgreSQL Documentation for information on archive_command.
- Refer to "B.1 WAL Mirroring Control Functions" for information on pgx_resume_wal_multiplexing.

8.2 Recovering from Data Corruption

If data in a disk is logically corrupted and the database does not operate properly, you can recover the database cluster to its state at the time of backup.

There are two methods of recovery:

- 8.2.1 Using WebAdmin
- 8.2.2 Using the pgx_rcvall Command

Note

- Back up the database cluster after recovering it. Backup deletes obsolete archive logs (transaction logs copied to the backup data storage destination), freeing up disk space and reducing the recovery time.
- If you recover data to a point in the past, a new time series (database update history) will start from that recovery point. When recovery is complete, the recovery point is the latest point in the new time series. When you subsequently recover data to the latest state, the database update is re-executed on the new time series.

8.2.1 Using WebAdmin

If using WebAdmin, recover the data to the point immediately prior to data corruption by using the backup data.

Refer to "8.1.1 Using WebAdmin" for details.

8.2.2 Using the pgx_rcvall Command

Recover the database cluster by specifying in the pgx_rcvall command the date and time of the backup you want to read from. Then re-execute the transaction as required to recover the data.

Follow the procedure below to recover the data storage disk.

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance. Refer to "2.1.2 Using Server Commands" for information on how to stop an instance.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

3. Confirm the backup date and time

   Execute the pgx_rcvall command to confirm the backup data saved in the backup data storage destination, and determine a date and time prior to data corruption.

   Specify the following values in the pg_rcvall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
   - Specify the backup storage directory in the -B option.
   - The -l option displays the backup data information.

   Example

   ```
   > pgx_rcvall -D /database/inst1 -B /backup/inst1 -l
   Date                    Status          Dir
   2015-05-20 10:00:00     COMPLETE        /backup/inst1/2015-05-20_10-00-00
   ```

4. Recover the keystore, and enable automatic opening of the keystore

   When the data in the database has been encrypted, restore the keystore to its state at the time of the database backup. Configure automatic opening of the keystore as necessary.

5. Recover the database cluster

   Use the pgx_rcvall command to recover the database cluster.

   Specify the following values in the pg_rcvall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
   - Specify the backup storage directory in the -B option.
   - Specify the recovery date and time in the -e option.

   Example

   In the following examples, "May 20, 2015 10:00:00" is specified as the recovery time.

   ```
   > pgx_rcvall -D /database/inst1 -B /backup/inst1 -e '2015-05-20 10:00:00'
   ```

Note

If recovery fails, remove the cause of the error in accordance with the displayed error message and then re-execute the pgx_rcvall command.

If the message "pgx_rcvall: an error occurred during recovery" is displayed, then the log recorded when recovery was executed is output after this message. The cause of the error is output in around the last fifteen lines of the log, so remove the cause of the error in accordance with the message and then re-execute the pgx_rcvall command.

The following message displayed during recovery is output as part of normal operation of pgx_rcvall command (therefore the user does not need not be concerned).

```
FATAL:  The database system is starting
```

6. Start the instance

   Start the instance. Refer to "2.1.2 Using Server Commands" for information on how to start an instance.

   If necessary, re-execute transaction processing from the specified recovery time, and then resume database operations.

   Note

   The pgx_rcvall command cannot accurately recover a hash index. If you are using a hash index, wait for the instance to start and then execute the REINDEX command for the appropriate index.

7. Resume applications

   Resume applications that are using the database.

See

Refer to "pgx_rcvall" in the Reference for information on the pgx_rcvall command.

8.3 Recovering from an Incorrect User Operation

This section describes how to recover database clusters when data has been corrupted due to erroneous user operations.

There are two methods of recovery:

- 8.3.1 Using WebAdmin
- 8.3.2 Using the pgx_rcvall Command

Note

- Back up the database cluster after recovering it. Backup deletes obsolete archive logs (transaction logs copied to the backup data storage destination), freeing up disk space and reducing the recovery time.
- If you recover data to a point in the past, a new time series (database update history) will start from that recovery point. When recovery is complete, the recovery point is the latest point in the new time series. When you subsequently recover data to the latest state, the database update is re-executed on the new time series.
- An effective restore point is one created on a time series for which you have made a backup. That is, if you recover data to a point in the past, you cannot use any restore points set after that recovery point. Therefore, once you manage to recover your target past data, make a backup.

8.3.1 Using WebAdmin

You can use WebAdmin to recover data to a backup point.

Follow the procedure below to recover the data in the data storage disk.

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance. Refer to "2.1.1 Using WebAdmin" for information on how to stop an instance.

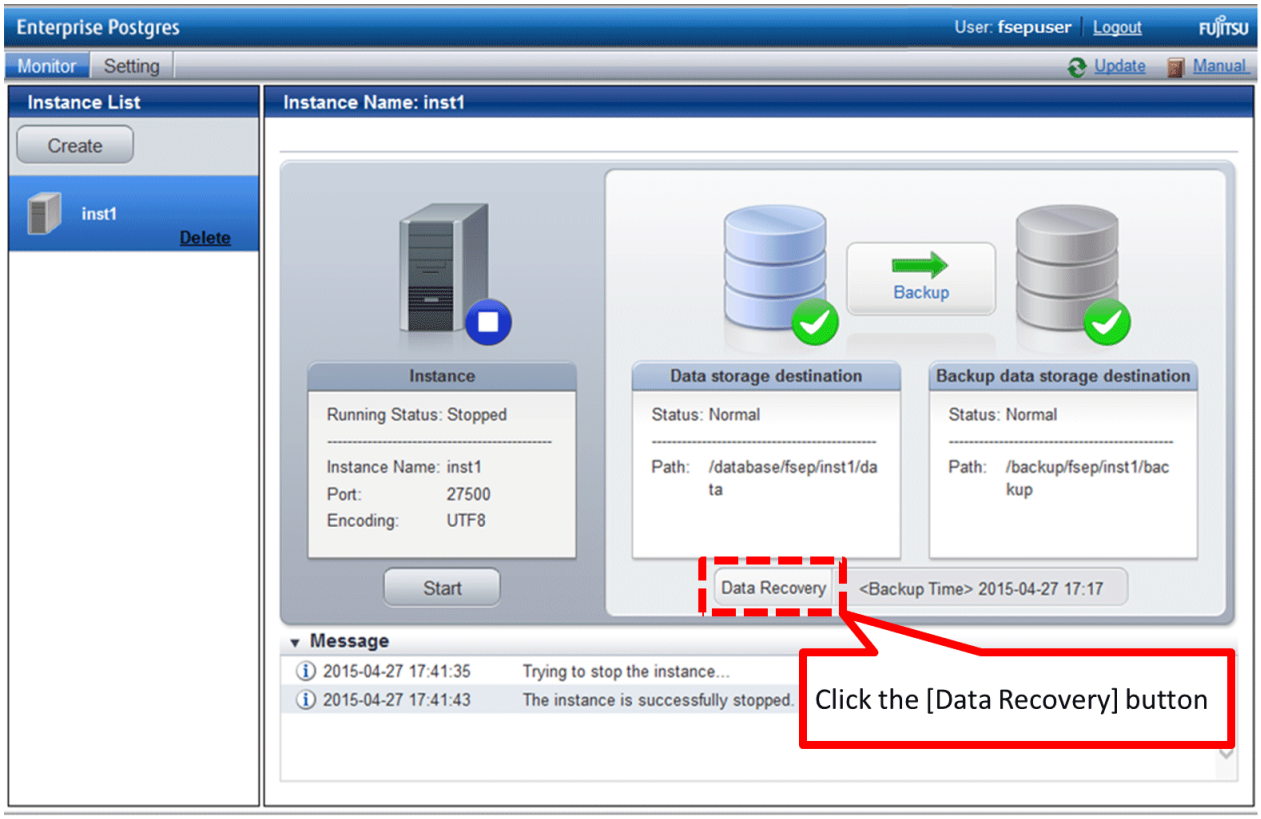
3. Recover the keystore, and enable automatic opening of the keystore

   Do the following if the data in the database has been encrypted:

   - Restore the keystore to its state at the time of the database backup.
   - Enable automatic opening of the keystore.

4. Recover the database cluster

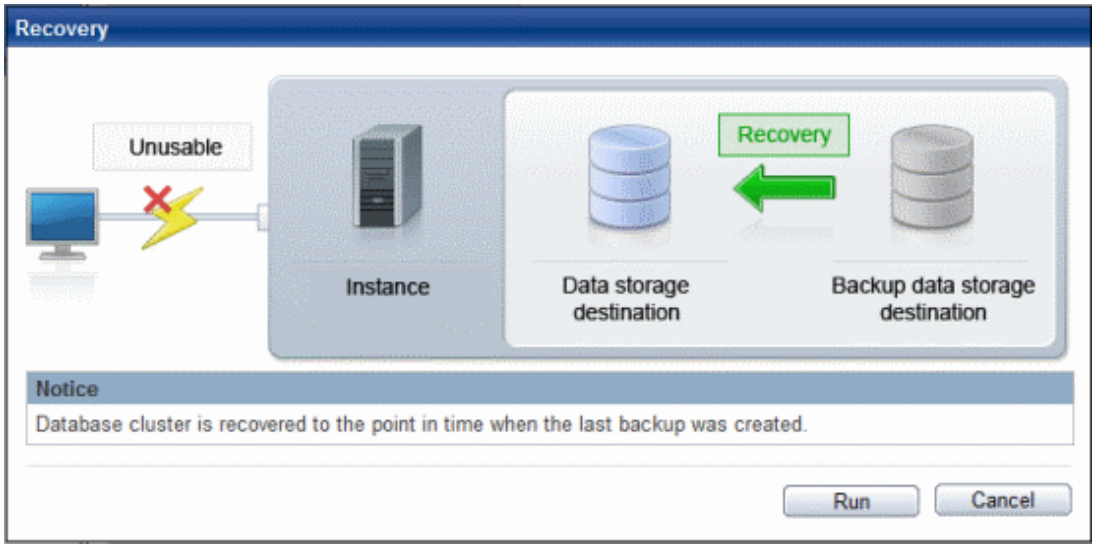
   Log in to WebAdmin, and in the [Monitor] window, click [Data Recovery].



5. Recover to the backup point

   In the [Recovery] dialog box that appears, click [Run].

   [Recovering] is displayed in the [Monitor] window, and recovery is performed. An instance is automatically started when recovery is successful.



Note

WebAdmin cannot accurately recover a hash index. If you are using a hash index, then after recovery, execute the REINDEX command for the appropriate index.

6. Resume database operations

   If necessary, re-execute transaction processing from the backup point to when an erroneous operation was performed, and then resume database operations.

8.3.2 Using the pgx_rcvall Command

The pgx_rcvall command recovers database clusters to the restore point created with the server command. Refer to "Setting a restore point" in "3.2.2 Using Server Commands" for information on how to create a restore point.

Follow the procedure below to recover the data in the data storage disk.

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance. Refer to "2.1.2 Using Server Commands" for information on how to stop an instance.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

3. Confirm the restore point

   Execute the pgx_rcvall command to confirm the backup data saved in the backup data storage destination, and use a restore point recorded in an arbitrary file, as explained in "3.2.2 Using Server Commands", to determine a restore point prior to the erroneous operation.

   Specify the following values in the pg_rcvall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
   - Specify the backup data storage destination in the -B option.
   - The -l option displays the backup data information.

   Example

   ```
   > pgx_rcvall -D /database/inst1 -B /backup/inst1 -l
   Date                    Status          Dir
   2015-05-01 10:00:00     COMPLETE        /backup/inst1/2015-05-01_10-00-00
   ```

4. Recover the keystore, and enable automatic opening of the keystore

   When the data in the database has been encrypted, restore the keystore to its state at the time of the database backup. Configure automatic opening of the keystore as necessary.

5. Recover the database cluster

   Use the pgx_rcvall command to recover the database cluster.

   Specify the following values in the pg_rcvall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
   - Specify the backup data storage destination in the -B option.
   - The -n option recovers the data to the specified restore point.

   Example

   The following example executes the pgx_rcvall command with the restore point "batch_20150503_1".

```
> pgx_rcvall -D /database/inst1 -B /backup/inst1 -n batch_20150503_1
```

Note

If recovery fails, remove the cause of the error in accordance with the displayed error message and then re-execute the pgx_rcvall command.

If the message "pgx_rcvall: an error occurred during recovery" is displayed, then the log recorded when recovery was executed is output after this message. The cause of the error is output in around the last fifteen lines of the log, so remove the cause of the error in accordance with the message and then re-execute the pgx_rcvall command.

The following message displayed during recovery is output as part of normal operation of pgx_rcvall (therefore the user does not need not be concerned).

```
FATAL: The database system is starting
```

6. Start the instance

   Start the instance.

   Refer to "2.1.2 Using Server Commands" for information on how to start an instance.

   Note

   The pgx_rcvall command cannot accurately recover a hash index. If you are using a hash index, wait for the instance to start and then execute the REINDEX command for the appropriate index.

7. Restart operation of the database

   If necessary, re-execute transaction processing from the specified recovery time to the point when an erroneous operation was performed, and then resume database operations.

See

Refer to "pgx_rcvall" in the Reference for information on the pgx_rcvall command.

8.4 Actions in Response to an Application Error

If there is a connection from a client that has been in the waiting state for an extended period, you can minimize performance degradation of the database by closing the problematic connection.

The following methods are available for identifying a connection to be closed:

- view(pg_stat_activity) (refer to "8.4.1 When using the view (pg_stat_activity)")
- ps command (refer to "8.4.2 Using the ps Command")
- pgAdmin (refer to "8.4.3 Using pgAdmin")

Use the system management function (pg_terminate_backend) to disconnect connections.

8.4.1 When using the view (pg_stat_activity)

When using the view (pg_stat_activity), follow the procedure below to close a connection.

1. Use psql command to connect to the postgres database.

```
> psql postgres
psql (9.4.3)
Type "help" for help.
```

2. Close connections from clients that have been in the waiting state for an extended period.

   Use pg_terminate_backend() to close connections that have been trying to connect for an extended period.

   However, when considering continued compatibility of applications, do not reference or use system catalogs and functions directly in SQL statements. Refer to " Notes on Application Compatibility" in the Application Development Guide for details.

   Example

   The following example closes connections where the client has been in the waiting state for at least 60 minutes.

```
select pid,usename,application_name,client_hostname,pg_terminate_backend(pid) from
pg_stat_activity where state='idle in transaction' and current_timestamp > cast(query_start +
interval '60 minutes' as timestamp);
-[ RECORD 1 ]--------+---------------
pid                  | 4684
username             | fsepuser
application_name     | apl1
client_addr          | 192.11.11.1
pg_terminate_backend | t
```

See

- Refer to "System Administration Functions" under "The SQL Language" in the PostgreSQL Documentation for information on pg_terminate_backend.
- Refer to "Notes on Application Compatibility" in the Application Development Guide for information on how to maintain application compatibility.

8.4.2 Using the ps Command

Follow the procedure below to close a connection using a standard Unix tool (ps command).

1. Execute the ps command.

```
> ps axwfo user,pid,ppid,tty,command | grep postgres
fsepuser 19174 18027 pts/1                \_ grep postgres
fsepuser 20517     1 ?        /opt/fsepserver64/bin/postgres -D /disk01/data
fsepuser 20518 20517 ?         \_ postgres: logger process
fsepuser 20520 20517 ?         \_ postgres: checkpointer process
fsepuser 20521 20517 ?         \_ postgres: writer process
fsepuser 20522 20517 ?         \_ postgres: wal writer process
fsepuser 20523 20517 ?         \_ postgres: autovacuum launcher process
fsepuser 20524 20517 ?         \_ postgres: archiver process
fsepuser 20525 20517 ?         \_ postgres: stats collector process
fsepuser 18673 20517 ?         \_ postgres: fsepuser postgres 192.168.100.1(49448) idle
fsepuser 16643 20517 ?         \_ postgres: fsepuser db01 192.168.100.11(49449) UPDATE waiting
fsepuser 16644 20517 ?       \_ postgres: fsepuser db01 192.168.100.12(49450) idle in transaction
```

   Process ID 16643 may be a connection that was established a considerable time ago by the UPDATE statement, or a connection that has occupied resources (waiting).

2. Close connections from clients that have been in the waiting state for an extended period.

   Use pg_terminate_backend() to close the connection with the process ID identified in step 1 above.

   However, when considering continued compatibility of applications, do not reference or use system catalogs and functions directly in SQL statements.

```
postgres=# SELECT pg_terminate_backend (16643);
 pg_terminate_backend
----------------------
```

```
t
(1 row)
```

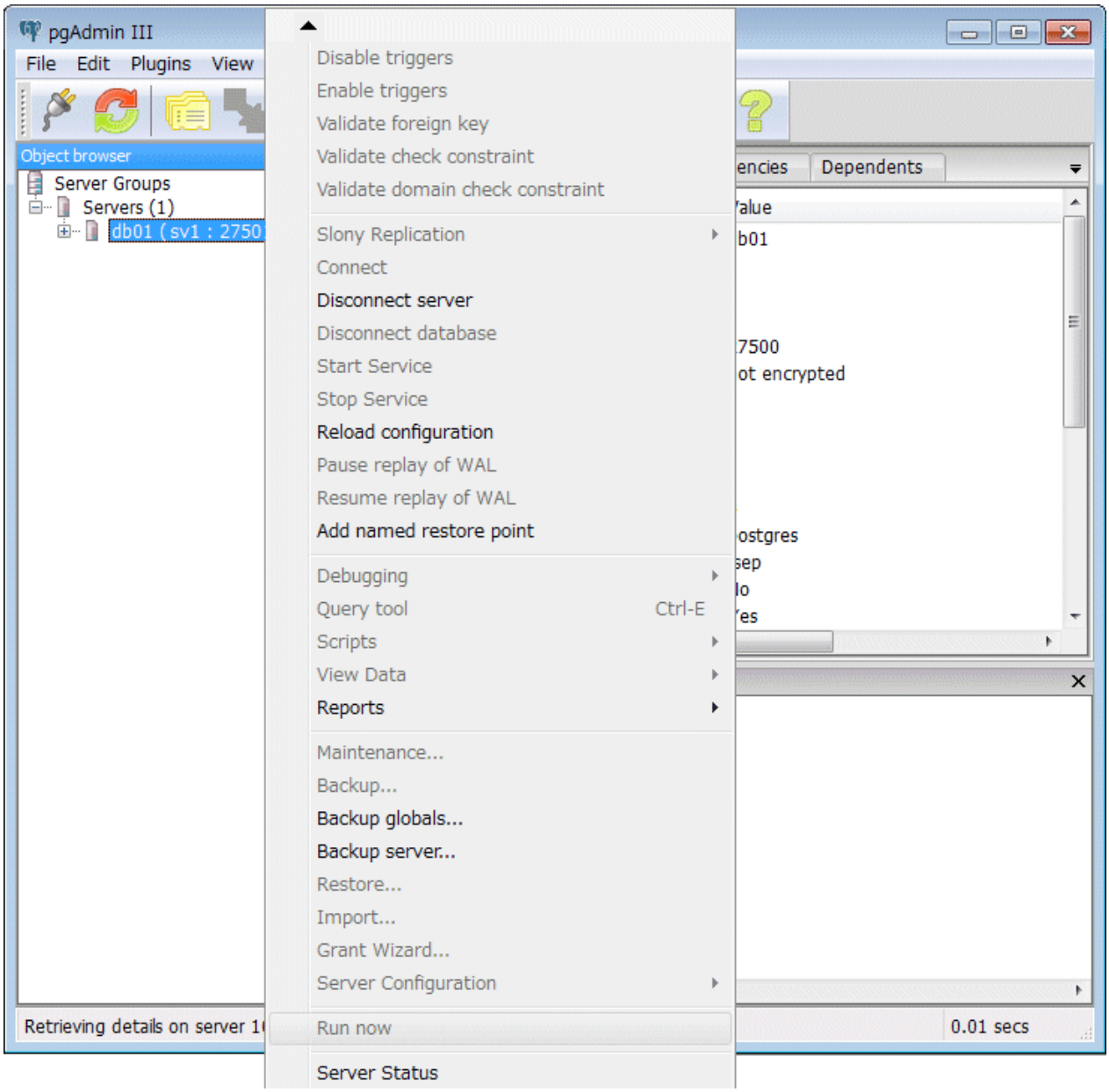
See

- Refer to "System Administration Functions" under "The SQL Language" in the PostgreSQL Documentation for information on pg_terminate_backend.
- Refer to "Notes on Application Compatibility" in the Application Development Guide for information on how to maintain application compatibility.

8.4.3 Using pgAdmin

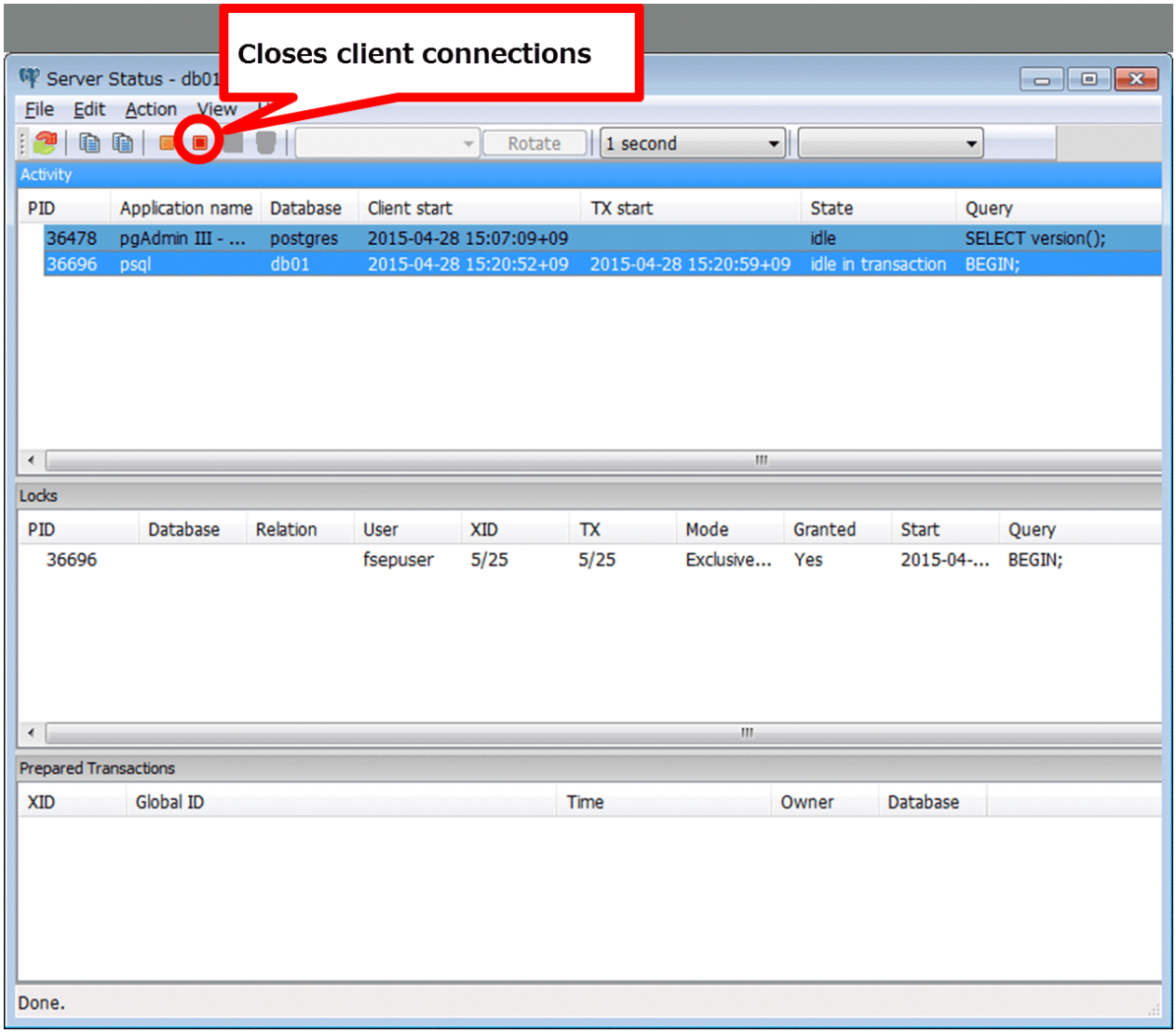
If using pgAdmin, follow the procedure below to close connections.

1. From the [Tools] menu in pgAdmin, click [Server Status].



2. Close client connections that have been in the waiting state for an extended period.

   From the transaction start time displayed under [TX Start], select connections that have been in the waiting state for an extended period. Then click the red square button to close the connections.



8.5 Actions in Response to an Access Error

If access is denied, grant privileges allowing the instance administrator to operate the following directories and then re-execute operations. Also, refer to the system log and the server log, and confirm that the file system has not been mounted as read-only due to a disk error. If the file system has been mounted as read-only, mount it properly and then re-execute operations.

- Data storage destination
- Tablespace storage destination
- Transaction log storage destination
- Backup data storage destination

See

Refer to "Preparing Directories to Deploy Resources" under "Setup" in the Installation and Setup Guide for Server for information on the privileges required for the directory.

8.6 Actions in Response to Insufficient Space on the Data Storage Destination

If the data storage destination runs out of space, check if the disk contains any unnecessary files and delete them so that operations can continue.

If deleting unnecessary files does not solve the problem, you must migrate data to a disk with larger capacity.

There are two methods of migrating data:

- 8.6.1 Using a Tablespace
- 8.6.2 Replacing the Disk with a Larger Capacity Disk

8.6.1 Using a Tablespace

Enterprise Postgres enables you to use a tablespace to change the storage destination of database objects, such as tables and indexes, to a different disk.

The procedure is as follows:

1. Create a tablespace

   Use the CREATE TABLESPACE command to create a new tablespace in the prepared disk.

2. Modify the tablespace

   Use the ALTER TABLE command to modify tables for the newly defined tablespace.

See

Refer to "SQL Commands" under "Reference" in the PostgreSQL Documentation for information on the CREATE TABLESPACE command and ALTER TABLE command.

8.6.2 Replacing the Disk with a Larger Capacity Disk

Before replacing the disk with a larger capacity disk, migrate resources at the data storage destination using the backup and recovery features.

There are two methods of performing backup and recovery:

- 8.6.2.1 Using WebAdmin
- 8.6.2.2 Using Server Commands

The following sections describe procedures that use each of these methods to replace the disk and migrate resources at the data storage destination.

Note

- Before replacing the disk, stop applications and instances that are using the database.
- It is recommended that you back up the database cluster following recovery. Backup deletes obsolete archive logs (transaction logs copied to the backup data storage destination), freeing up disk space and reducing the recovery time.

8.6.2.1 Using WebAdmin

Follow the procedure below to replace the disk and migrate resources at the data storage destination by using WebAdmin.

1. Back up files

   If the disk at the data storage destination contains any required files, back up the files. It is not necessary to back up the data storage destination.

2. Stop applications

   Stop applications that are using the database.

3. Back up the database cluster

   Back up the latest resources at the data storage destination. Refer to "3.2.1 Using WebAdmin" for details.

4. Stop the instance

   Stop the instance. Refer to "2.1.1 Using WebAdmin" for information on how to stop an instance.

5. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

6. Recover the database cluster

   Log in to WebAdmin, and perform recovery operations. Refer to steps 4 ("Create a tablespace directory ") to 7 ("Run recovery") under "If failure occurred in the data storage disk or the transaction log storage disk" in "8.1.1 Using WebAdmin" for information on the procedure. An instance is automatically started when recovery is successful.

7. Resume applications

   Resume applications that are using the database.

8. Restore the files

   Restore the files backed up in step 1.

8.6.2.2 Using Server Commands

Follow the procedure below to replace the disk and migrate resources at the data storage destination by using server commands.

1. Back up files

   If the disk at the data storage destination contains any required files, back up the files. It is not necessary to back up the data storage destination.

2. Stop applications

   Stop applications that are using the database.

3. Back up the database cluster

   Back up the latest resources at the data storage destination. Refer to "3.2.2 Using Server Commands" for details.

4. Stop the instance

   After backup is complete, stop the instance. Refer to "2.1.2 Using Server Commands" for information on how to stop an instance.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

5. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

6. Create a data storage destination

   Create a data storage destination. If a tablespace was defined, also create a directory for it.

   Example

   ```
   $ mkdir /database/inst1
   $ chown fsepuser:fsepuser /database/inst1
   $ chmod 700 /database/inst1
   ```

7. Recover the keystore, and enable automatic opening of the keystore

   When the data in the database has been encrypted, restore the keystore to its state at the time of the database backup. Configure automatic opening of the keystore as necessary.

8. Recover the database cluster

   Use the pgx_rcvall command to recover the database cluster.

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
   - Specify the backup storage directory in the -B option.

   Example

```
> pgx_rcvall -D /database/inst1 -B /backup/inst1
```

   Note

   If recovery fails, remove the cause of the error in accordance with the displayed error message and then re-execute the pgx_rcvall command.

   If the message "pgx_rcvall: an error occurred during recovery" is displayed, then the log recorded when recovery was executed is output after this message. The cause of the error is output in around the last fifteen lines of the log, so remove the cause of the error in accordance with the message and then re-execute the pgx_rcvall command.

   The following message displayed during recovery is output as part of normal operation of pgx_rcvall (therefore the user does not need not be concerned).

```
FATAL:  The database system is starting
```

   See

   Refer to "pgx_rcvall" in the Reference for information on the pgx_rcvall command.

9. Start the instance

   Start the instance.

   Refer to "2.1.2 Using Server Commands" for information on how to start an instance.

   Note

   The pgx_rcvall command cannot accurately recover a hash index. If you are using a hash index, wait for the pgx_rcvall command to end and then execute the REINDEX command for the appropriate index.

10. Resume applications

    Resume applications that are using the database.

11. Restore files

    Restore the files backed up in step 1.

8.7 Actions in Response to Insufficient Space on the Backup Data Storage Destination

If space runs out on the backup data storage destination, check if the disk contains any unnecessary files and delete them, and then make a backup as required.

If deleting unnecessary files does not solve the problem, take the following action:

- 8.7.1 Temporarily Saving Backup Data
- 8.7.2 Replacing the Disk with a Larger Capacity Disk

8.7.1 Temporarily Saving Backup Data

This method involves temporarily moving backup data to a different directory, saving it there, and securing disk space on the backup data storage destination so that a backup can be made normally.

Use this method if you need time to prepare a larger capacity disk.

If space runs out on the backup data storage destination, archive logs can no longer be stored in the backup data storage destination. As a result, transaction logs continue to accumulate in the data storage destination or the transaction log storage destination.

If action is not taken soon, the transaction log storage destination will become full, and operations may not be able to continue.

To prevent this, secure space in the backup data storage destination, so that archive logs can be stored.

There are two methods of taking action:

- 8.7.1.1 Using WebAdmin
- 8.7.1.2 Using Server Commands

8.7.1.1 Using WebAdmin

Follow the procedure below to recover the backup data storage disk.

1. Temporarily save backup data

   Move backup data to a different directory and temporarily save it, and secure space in the backup data storage destination directory.

   The reason for saving the backup data is so that the data in the data storage destination can be recovered even if it is corrupted before you perform recovery. If there is no disk at the save destination and you consider that there is no risk of corruption at the data storage destination, delete the backup data.

   The following example saves backup data from the backup data storage destination directory (/backup/inst1) under /mnt/usb/backup.

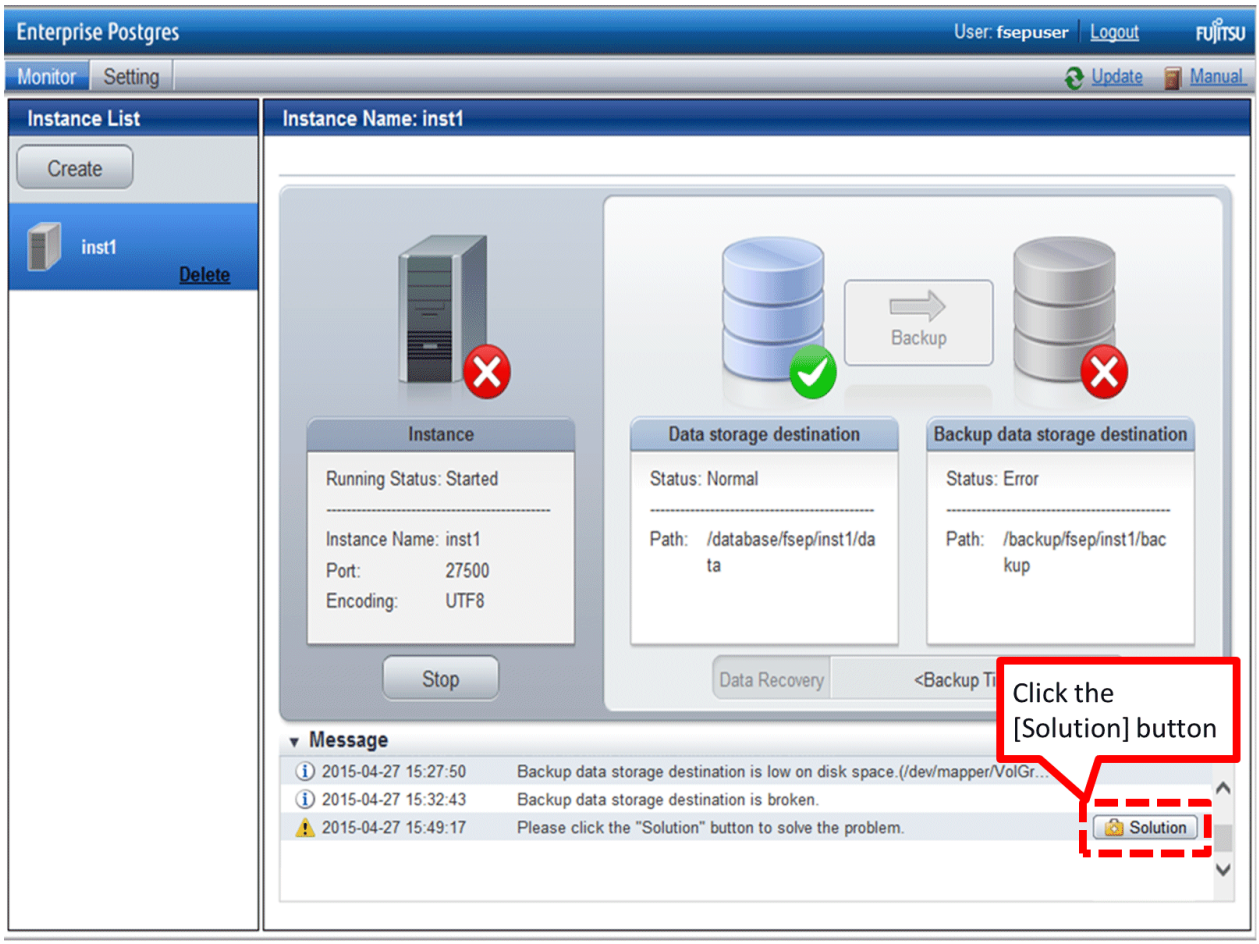
   Example

   ```
   > mkdir /mnt/usb/backup/
   > mv /backup/inst1/* /mnt/usb/backup/
   ```

2. Recover backup data

   Log in to WebAdmin and start recovering backup data.

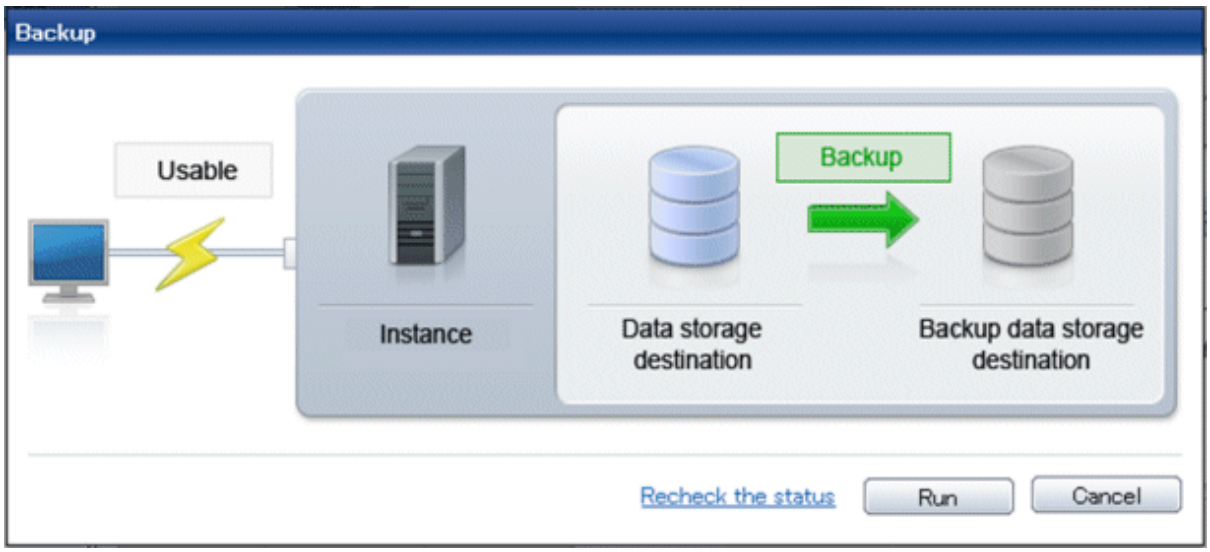
In the [Monitor] window, click [Solution] for the error message.



The above screen is displayed if moving and saving backup data causes WebAdmin to detect that the content at the backup data storage destination is missing. If you use another method to save the data and no abnormality is detected, click [->] next to the "Backup" caption in the [Monitor] menu window.

3. Run backup

Perform the backup to enable the recovery of backup data. In the [Backup] dialog box displayed, click [Run]. [Backuping] is displayed in the [Monitor] window and the backup is performed. An instance is automatically activated when backup is performed.



4. Delete temporarily saved backup data

   If backup completes normally, the temporarily saved backup data becomes unnecessary and is deleted.

   The following example deletes backup data that was temporarily saved in /mnt/usb.

   Example

```
> rm -rf /mnt/usb/backup
```

8.7.1.2 Using Server Commands

The following describes the procedure for recovering the backup storage disk.

There are two methods of taking action:

- Performing recovery while the instance is active
- Stopping the instance before performing recovery

The following table shows the different steps to be performed depending on whether you stop the instance.

No	Step	Instance stopped	
		No	Yes
1	Stop transaction log mirroring	Y	N
2	Stop output of archive logs	Y	N
3	Stop applications	N	Y
4	Stop the instance	N	Y
5	Temporarily save backup data	Y	Y
6	Resume output of archive logs	Y	N
7	Resume transaction log mirroring	Y	N
8	Start an instance	N	Y
9	Run backup	Y	Y
10	Resume applications	N	Y
11	Delete temporarily saved backup data	Y	Y

Y: Required
N: Not required

The procedure is as follows:

Performing recovery while the instance is active

1. Stop transaction log mirroring

   Stop transaction log mirroring.

```
postgres=# SELECT pgx_pause_wal_multiplexing();
LOG:  multiplexing of transaction log files has been stopped
pgx_pause_wal_multiplexing
---------------------------

(1 row)
```

2. Stop output of archive logs

   Transaction logs may accumulate during replacement of backup storage disk, and if the data storage disk or the transaction log storage disk becomes full, there is a risk that operations may not be able to continue.

To prevent this, use the following methods to stop output of archive logs.

- Changing the archive_command parameter

  Specify a command that will surely complete normally, such as "echo skipped archiving WAL file %f" or "/bin/true", so that archive logs will be regarded as having been output.

  If you specify echo, a message is output to the server log, so it may be used as a reference when you conduct investigations.

- Reloading the configuration file

  Run the pg_ctl reload command or the pg_reload_conf SQL function.

If you simply want to stop output of errors without the risk that operations will not be able to continue, specify an empty string ('') in archive_command and reload the configuration file.

3. Temporarily save backup data

   Move backup data to a different directory and temporarily save it, and secure space in the backup data storage destination directory.

   The reason for saving the backup data is so that the data in the data storage destination can be recovered even if it is corrupted before you perform the next step. If there is no disk at the save destination and you consider that there is no risk of corruption at the data storage destination, delete the backup data.

   The following example saves backup data from the backup data storage destination directory (/backup/inst1) under /mnt/usb/backup.

   Example

   ```
   > mkdir /mnt/usb/backup/
   > mv /backup/inst1/* /mnt/usb/backup/
   ```

4. Resume output of archive logs

   Return the archive_command setting to its original value, and reload the configuration file.

5. Resume transaction log mirroring

   Execute the pgx_resume_wal_multiplexing SQL function.

   Example

   ```
   SELECT pgx_resume_wal_multiplexing()
   ```

6. Run backup

   Use the pgx_dmpall command to back up the database cluster.

   Specify the following option in the pgx_dmpall command:

   - Specify the directory of the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

     Example

     ```
     > pgx_dmpall -D /database/inst1
     ```

7. Delete temporarily saved backup data

   If backup completes normally, the temporarily saved backup data becomes unnecessary and is deleted.

   The following example deletes backup data that was temporarily saved in /mnt/usb.

   Example

   ```
   > rm -rf /mnt/usb/backup
   ```

If an instance has been stopped

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance. Refer to "2.1.2 Using Server Commands" for details.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

3. Temporarily save backup data

   Move backup data to a different directory and temporarily save it, and secure space in the backup data storage destination directory.

   The reason for saving the backup data is so that the data in the data storage destination can be recovered even if it is corrupted before you perform recovery. If there is no disk at the save destination and you consider that there is no risk of corruption at the data storage destination, delete the backup data.

   The following example saves backup data from the backup data storage destination directory (/backup/inst1) under /mnt/usb/backup.

   Example

   ```
   > mkdir /mnt/usb/backup/
   > mv /backup/inst1/* /mnt/usb/backup/
   ```

4. Start the instance

   Start the instance. Refer to "2.1.2 Using Server Commands" for information on how to start an instance.

5. Run backup

   Use the pgx_dmpall command to back up the database cluster.

   Specify the following value in the pgx_dmpall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

     Example

     ```
     > pgx_dmpall -D /database/inst1
     ```

6. Resume applications

   Resume applications that are using the database.

7. Delete temporarily saved backup data

   If backup completes normally, the temporarily saved backup data becomes unnecessary and is deleted.

   The following example deletes backup data that was temporarily saved in /mnt/usb.

   Example

   ```
   > rm -rf /mnt/usb/backup
   ```

See

- Refer to "pgx_rcvall" and "pgx_dmpall" in the Reference for information on the pgx_rcvall command and pgx_dmpall command.
- Refer to "Write Ahead Log" under "Server Administration" in the PostgreSQL Documentation for information on archive_command.
- Refer to "B.1 WAL Mirroring Control Functions" for information on the pgx_is_wal_multiplexing_paused and pgx_resume_wal_multiplexing.

8.7.2 Replacing the Disk with a Larger Capacity Disk

This method involves replacing the disk at the backup data storage destination with a larger capacity disk, so that it does not run out of free space again. After replacing the disk, back up data to obtain a proper backup.

There are two methods of performing backup:

- 8.7.2.1 Using WebAdmin
- 8.7.2.2 Using Server Commands

Note

Before replacing the disk, stop applications that are using the database.

8.7.2.1 Using WebAdmin

Follow the procedure below to recover the backup storage disk.

1. Back up files

   If the disk at the backup data storage destination contains any required files, back up the files. It is not necessary to back up the backup data storage destination.

2. Temporarily save backup data

   Save the backup data to a different directory.

   The reason for saving the backup data is so that the data in the data storage destination can be recovered even if it is corrupted before you perform the next step. If there is no disk at the save destination and you consider that there is no risk of corruption at the data storage destination, delete the backup data.

   The following example saves backup data from the backup data storage destination directory (/backup/inst1) under /mnt/usb/backup.

   Example

   ```
   > mkdir /mnt/usb/backup/
   > mv /backup/inst1/* /mnt/usb/backup/
   ```

3. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

4. Run backup

   Log in to WebAdmin, and perform recovery operations. Refer to steps 2 ("Recover the backup data") and 3 ("Run backup") under "If failure occurred on the backup storage disk" in "8.1.1 Using WebAdmin".

5. Restore files

   Restore the files backed up in step 1.

6. Delete temporarily saved backup data

   If backup completes normally, the temporarily saved backup data becomes unnecessary and is deleted.

   The following example deletes backup data that was temporarily saved in /mnt/usb.

   Example

   ```
   > rm -rf /mnt/usb/backup
   ```

8.7.2.2 Using Server Commands

The procedure for recovering the backup data storage disk is described below.

There are two methods of taking action:

- Performing recovery while the instance is active
- Stopping the instance before performing recovery

The following table shows the different steps to be performed depending on whether you stop the instance.

No	Step	Instance stopped	
		No	Yes
1	Back up files	Y	Y
2	Temporarily save backup data	Y	Y
3	Confirm that transaction log mirroring has stopped	Y	N
4	Stop output of archive logs	Y	N
5	Stop applications	N	Y
6	Stop the instance	N	Y
7	Replace with a larger capacity disk	Y	Y
8	Create a backup storage directory	Y	Y
9	Resume output of archive logs	Y	N
10	Resume transaction log mirroring	Y	N
11	Start the instance	N	Y
12	Run backup	Y	Y
13	Resume applications	N	Y
14	Restore files	Y	Y
15	Delete temporarily saved backup data	Y	Y

Y: Required
N: Not required

The procedure is as follows:

If an instance has not been stopped

1. Back up files

   If the disk at the backup data storage destination contains any required files, back up the files. It is not necessary to back up the backup data storage destination.

2. Temporarily save backup data

   Save the backup data to a different directory.

   The reason for saving the backup data is so that the data in the data storage destination can be recovered even if it is corrupted before you perform the next step. If there is no disk at the save destination and you consider that there is no risk of corruption at the data storage destination, delete the backup data.

   The following example saves backup data from the backup data storage destination directory (/backup/inst1) under /mnt/usb/backup.

   Example

   ```
   > mkdir /mnt/usb/backup/
   > mv /backup/inst1/* /mnt/usb/backup/
   ```

3. Confirm that transaction log mirroring has stopped

   Use the following SQL function to confirm that transaction log mirroring has stopped.

   ```
    postgres=# SELECT pgx_is_wal_multiplexing_paused();
    pgx_is_wal_multiplexing_paused
   --------------------
    t
   (1 row)
   ```

If transaction log mirroring has not stopped, then stop it using the following SQL function.

```
postgres=# SELECT pgx_pause_wal_multiplexing();
LOG:  multiplexing of transaction log files has been stopped
pgx_pause_wal_multiplexing
---------------------------

(1 row)
```

4. Stop output of archive logs

   Transaction logs may accumulate during replacement of backup storage disk, and if the data storage destination disk or the transaction log storage destination disk becomes full, there is a risk that operations may not be able to continue.

   To prevent this, use the following methods to stop output of archive logs.

   - Changing the archive_command parameter

     Specify a command that will surely complete normally, such as "echo skipped archiving WAL file %f" or "/bin/true", so that archive logs will be regarded as having been output.

     If you specify echo, a message is output to the server log, so it may be used as a reference when you conduct investigations.

   - Reloading the configuration file

     Run the pg_ctl reload command or the pg_reload_conf SQL function.

   If you simply want to stop output of errors without the risk that operations will not be able to continue, specify an empty string ('') in archive_command and reload the configuration file.

5. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

6. Create a backup data storage destination

   Create a backup data storage destination.

   Example

   ```
   # mkdir /backup/inst1
   # chown fsepuser:fsepuser /backup/inst1
   # chmod 700 /backup/inst1
   ```

   Refer to "3.2.2 Using Server Commands" for details.

7. Resume output of archive logs

   Return the archive_command setting to its original value, and reload the configuration file.

8. Resume transaction log mirroring

   Execute the pgx_resume_wal_multiplexing SQL function.

   Example

   ```
   SELECT pgx_resume_wal_multiplexing()
   ```

9. Run backup

   Use the pgx_dmpall command to back up the database cluster.

   Specify the following value in the pgx_dmpall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

   Example

   ```
   > pgx_dmpall -D /database/inst1
   ```

10. Restore files

    Restore the files backed up in step 1.

11. Delete temporarily saved backup data

    If backup completes normally, the temporarily saved backup data becomes unnecessary and is deleted.

    The following example deletes backup data that was temporarily saved in /mnt/usb.

    Example

    ```
    > rm -rf /mnt/usb/backup
    ```

If an instance has been stopped

1. Back up files

    If the disk at the backup data storage destination contains any required files, back up the files. It is not necessary to back up the backup data storage destination.

2. Temporarily save backup data

    Save the backup data to a different directory.

    The reason for saving the backup data is so that the data in the data storage destination can be recovered even if it is corrupted before you perform the next step. If there is no disk at the save destination and you consider that there is no risk of corruption at the data storage destination, delete the backup data.

    The following example saves backup data from the backup data storage destination directory (/backup/inst1) under /mnt/usb/backup.

    Example

    ```
    > mkdir /mnt/usb/backup/
    > mv /backup/inst1/* /mnt/usb/backup/
    ```

3. Stop applications

    Stop applications that are using the database.

4. Stop the instance

    Stop the instance. Refer to "2.1.2 Using Server Commands" for information on how to stop an instance.

    If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

5. Replace with a larger capacity disk

    Replace the disk. Then, recover the volume configuration information.

6. Create a backup data storage destination

    Create a backup data storage destination.

    Example

    ```
    # mkdir /backup/inst1
    # chown fsepuser:fsepuser /backup/inst1
    # chmod 700 /backup/inst1
    ```

    Refer to "3.2.2 Using Server Commands" for details.

7. Start the instance

    Start the instance. Refer to "2.1.2 Using Server Commands" for information on how to start an instance.

8. Run backup

    Use the pgx_dmpall command to back up the database cluster.

    Specify the following value in the pgx_dmpall command:

- Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

Example

```
> pgx_dmpall -D /database/inst1
```

9. Resume applications

   Resume applications that are using the database.

10. Restore files

    Restore the files backed up in step 1.

11. Delete temporarily saved backup data

    If backup completes normally, the temporarily saved backup data becomes unnecessary and is deleted.

    The following example deletes backup data that was temporarily saved in /mnt/usb.

    Example

    ```
    > rm -rf /mnt/usb/backup
    ```

See

- Refer to "pgx_rcvall" and "pgx_dmpall" in the Reference for information on the pgx_rcvall command and pgx_dmpall command.
- Refer to "Write Ahead Log" under "Server Administration" in the PostgreSQL Documentation for information on archive_command.
- Refer to "B.1 WAL Mirroring Control Functions" for information on the pgx_is_wal_multiplexing_paused and pgx_resume_wal_multiplexing.

8.8 Actions in Response to Insufficient Space on the Transaction Log Storage Destination

If the transaction log storage destination runs out of space, check if the disk contains any unnecessary files and delete them so that operations can continue.

If deleting unnecessary files does not solve the problem, you must migrate data to a disk with larger capacity.

8.8.1 Replacing the Disk with a Larger Capacity Disk

Before replacing the disk with a larger capacity disk, migrate resources at the transaction log storage destination using the backup and recovery features.

There are two methods of performing backup and recovery:

- 8.8.1.1 Using WebAdmin
- 8.8.1.2 Using Server Commands

The following sections describe procedures that use each of these methods to replace the disk and migrate resources at the transaction log storage destination.

Note

- Before replacing the disk, stop applications that are using the database.
- It is recommended that you back up the database cluster following recovery. Backup deletes obsolete archive logs (transaction logs copied to the backup data storage destination), freeing up disk space and reducing the recovery time.

8.8.1.1 Using WebAdmin

Follow the procedure below to replace the disk and migrate resources at the transaction log storage destination by using WebAdmin.

1. Back up files

   If the disk at the transaction log storage destination contains any required files, back up the files. It is not necessary to back up the transaction log storage destination.

2. Back up the database cluster

   Back up the latest data storage destination resources and transaction log storage destination resources (refer to "3.2.1 Using WebAdmin" for details).

3. Stop applications

   Stop applications that are using the database.

4. Stop the instance

   Stop the instance. Refer to "2.1.1 Using WebAdmin" for information on how to stop an instance. WebAdmin automatically stops instances if recovery of the database cluster is performed without stopping the instance.

5. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

6. Create a tablespace directory

   If a tablespace was defined after backing up, create a directory for it.

7. Recover the keystore, and enable automatic opening of the keystore

   Do the following if the data in the database has been encrypted:

   - Restore the keystore to its state at the time of the database backup.
   - Enable automatic opening of the keystore.

8. Recover the database cluster

   Log in to WebAdmin, and perform recovery operations. Refer to steps 4 ("Create a tablespace directory ") to 7 ("Run Recovery") under " If failure occurred in the data storage disk or the transaction log storage disk " in "8.1.1 Using WebAdmin" for information on the procedure. An instance is automatically started when recovery is successful.

9. Resume applications

   Resume applications that are using the database.

10. Restore files

    Restore the files backed up in step 1.

8.8.1.2 Using Server Commands

Follow the procedure below to replace the disk and migrate resources at the transaction log storage destination by using server commands.

1. Back up files

   If the disk at the transaction log storage destination contains any required files, back up the files. It is not necessary to back up the transaction log storage destination.

2. Back up the database cluster

   Use server commands to back up the latest data storage destination resources and transaction log storage destination resources. Refer to "3.2.2 Using Server Commands" for information on how to perform backup.

3. Stop applications

   Stop applications that are using the database.

4. Stop the instance

   After backup is complete, stop the instance. Refer to "2.1.2 Using Server Commands" for information on how to stop an instance.

If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

5. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

6. Create a transaction log storage destination

   Create a transaction log storage destination. If a tablespace was defined, also create a directory for it.

   Example

   ```
   # mkdir /tranlog/inst1
   # chown fsepuser:fsepuser /tranlog/inst1
   # chmod 700 /tranlog/inst1
   ```

7. Recover the keystore, and enable automatic opening of the keystore

   When the data in the database has been encrypted, restore the keystore to its state at the time of the database backup. Configure automatic opening of the keystore as necessary.

8. Recover the database cluster

   Use the pgx_rcvall command to recover the database cluster.

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
   - Specify the backup storage directory in the -B option.

   Example

   ```
   > pgx_rcvall -D /database/inst1 -B /backup/inst1
   ```

   Note

   If recovery fails, remove the cause of the error in accordance with the displayed error message and then re-execute the pgx_rcvall command.

   If the message "pgx_rcvall: an error occurred during recovery" is displayed, then the log recorded when recovery was executed is output after this message. The cause of the error is output in around the last fifteen lines of the log, so remove the cause of the error in accordance with the message and then re-execute the pgx_rcvall command.

   The following message displayed during recovery is output as part of normal operation of pgx_rcvall command (therefore the user does not need not be concerned).

   ```
   FATAL: The database system is starting
   ```

   See

   Refer to "pgx_rcvall" in the Reference for information on the pgx_rcvall command.

9. Start the instance

   Start the instance.

   Refer to "2.1.2 Using Server Commands" for information on how to start an instance.

   Note

   The pgx_rcvall command cannot accurately recover a hash index. If you are using a hash index, wait for the instance to start and then execute the REINDEX command for the appropriate index.

10. Resume applications

    Resume applications that are using the database.

11. Restore files

    Restore the files backed up in step 1.

8.9 Errors in More Than One Storage Disk

If an error occurs in the storage destination disks or resources are corrupted, determine the cause of the error from system logs and server logs and remove the cause.

If errors occur in either of the following combinations, you cannot recover the database.

Recreate the instance, and rebuild the runtime environment.

Data storage disk	Transaction log storage disk	Backup data storage disk
Error	-	Error
-	Error	Error

Refer to "Setup" in the Installation and Setup Guide for Server for information on how to create an instance and build the runtime environment.

8.10 Actions in Response to Instance Startup Failure

If an instance fails to start, refer to the system log and the server log, and determine the cause of the failure.

If using WebAdmin, remove the cause of the error. Then, click [Solution] and [Recheck the status] and confirm that the instance is in the normal state.

The following sections describe common causes of errors and the actions to take.

8.10.1 Errors in the Configuration File

If you have directly edited the configuration file using a text editor or changed the settings using WebAdmin, refer to the system log and the server log, confirm that no messages relating to the files below have been output.

- postgresql.conf
- pg_hba.conf

Refer to the following for information on the parameters in the configuration file:

- "Configuring Parameters" in the Installation and Setup Guide for Server
- "Appendix A Parameters"
- "Server Configuration" and "Client Authentication" under "Server Administration" in the PostgreSQL Documentation

8.10.2 Errors Caused by Power Failure or Mounting Issues

If mounting is cancelled after restarting the server, for example, because the disk device for each storage destination disk was not turned on, or because automatic mounting has not been set, then starting an instance will fail.

Refer to "8.13.2 Errors Caused by Power Failure or Mounting Issues", and take actions accordingly.

8.10.3 Errors Caused by Failure to Start PL/extJava

If an error occurs during instance startup because PL/extJava has failed to start, check the error messages output to the logs below and investigate the cause of the error by referring to "Messages Output by PL/extJava" in Messages. Refer to "Appendix G PL/extJava Log Information" for information on the location of these logs.

- System log
- Enterprise Postgres server log
- Container server log
- Container Java VM log
- Domain server log
- Domain Java VM log

Some common causes of this error (and their corresponding corrective actions) are listed below:

- A fault may have occurred on the disk where the domain root is placed, or the disk may have becomes corrupted by an incorrect operation.
  If that is the case, perform the following procedure to restore it

  If a backup of PL/extJava has been obtained

  Refer to "7.3.7.2 Restore Method" and restore PL/extJava accordingly.

  If a backup of PL/extJava has not been obtained

  Perform the following procedure.

  1. Stop applications

     Stop applications that are using the database.

  2. Stop the instance

     Stop the instance, refer to 2.1 Starting and Stopping an Instance" for details.

     If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

  3. Recover the failed disk

     If the disk is defective, replace the disk, and then recover the volume configuration information.

  4. Delete the domain root directory

  5. Change the settings in the postgresql.conf file

     Delete the following parameters from the postgresql.conf file.

     - plextjava.http_port
     - plextjava.start_command
     - plextjava.stop_command
     - plextjava.forcible_stop_command

  6. Delete the domain root information

     Delete the files listed below.

     - *Enterprise PostgresInstallDir*/java/etc/domain.conf
     - *Enterprise PostgresInstallDir*/java/etc/domain1_container.conf
     - /etc/init.d/FJSVpcmiisje6-*componentId*
     - /etc/rc.d/rc0.d/K00FJSVpcmiisje6-*componentId*

- /etc/rc.d/rc1.d/K00FJSVpcmiisje6-*componentId*
- /etc/rc.d/rc2.d/S99FJSVpcmiisje6-*componentId*
- /etc/rc.d/rc3.d/S99FJSVpcmiisje6-*componentId*
- /etc/rc.d/rc4.d/S99FJSVpcmiisje6-*componentId*
- /etc/rc.d/rc5.d/S99FJSVpcmiisje6-*componentId*
- /etc/rc.d/rc6.d/K00FJSVpcmiisje6-*componentId*

componentId has the following format:

```
STFFSEPDB + versionLevel + firstDigitOfArchitecture
```

- *versionLevel*: Version/level of Enterprise Postgres
- *firstDigitOfArchitecture: First digit of product architecture, that is "3" for the 32-bit version, and "6" for the 64-bit version*

For Enterprise Postgres version "0941" with 64-bit product architecture, componentId will be:.

```
STFFSEPDB09416
```

7. Recreate PL/extJava

   Recreate PL/extJava after executing the pgx_jadmin command for PL/extJava and storing the Java applications. Refer to "Chapter 7 Setting up and Operating PL/extJava " for information on creating PL/extJava.

8. Start the instance

   Refer to 2.1 Starting and Stopping an Instance" for information on how to start an instance.

   If the instance fails to stop, refer to "8.10 Actions in Response to Instance Startup Failure".

9. Resume applications

   Resume applications that are using the database.

- If a message was output indicating that the address cannot be assigned to the container server log or the domain server log, the specified port number may be in use by another server.
  If that is the case, change the port numbers and then restart the instance (refer to "7.2.1 Preparing Port Numbers" and "7.3.3.3 Changing Port Numbers" for details).

- If a message was output to the server log of Enterprise Postgres indicating that the Java EE DAS service has failed to start, but a corresponding message indicating the cause of the failure has not been output to the domain server log, the cached information may not be up to date.
  If that is the case, delete the file below and then restart the instance.

```
domainRoot/domains/domain1/osgi-cache/felix
```

- If a message was output indicating that there are PL/extJava processes already running, there may be processes still running after the PCMI service that monitors processes ended in an error.
  If that is the case, perform the following procedure to forcibly terminate the unnecessary processes, and then restart the instance.

1. End server instances(Java VM)

   Forcibly terminate any processes listed in the file below and that are still running, then delete the file (perform this step for all server instances(Java VM)).

   ```
   domainRoot/nodes/localhost-domain1/serverInstance(JavaVM)Name/config/pid
   ```

2. End Java EE DAS service processes

   Forcibly terminate any processes listed in the file below and that are still running, then delete the file.

```
domainRoot/domains/domain1/config/pid
```

3. End web server processes

   Forcibly terminate any processes listed in the file below and that are still running, then delete the file.

```
Enterprise PostgresInstallDir/java/ahs22/plextjava/domain1/logs/httpd.pid
```

4. End PCMI service processes

   Use the method below to check if there are PCMI service processes still running - if there are, forcibly terminate them, and then restart the instance.

```
ps -ef | grep "com.fujitsu.interstage.pcmi.PcmiMain domainRoot/pcmi/isje6"
```

8.10.4 Other Errors

This section describes the recovery procedure to be used if you cannot take any action or the instance cannot start even after you have referred to the system log and the server log.

There are two methods of recovery:

- 8.10.4.1 Using WebAdmin
- 8.10.4.2 Using Server Commands

Note that recovery will not be possible if there is an error at the backup data storage destination. If the problem cannot be resolved, contact Fujitsu technical support.

8.10.4.1 Using WebAdmin

Follow the procedure below to perform recovery.

1. Delete the data storage destination directory and the transaction log storage destination directory

   Back up the data storage destination directory and the transaction log storage destination directory before deleting them.

2. Reconfirm the status

   Log in to WebAdmin, and in the [Monitor] window, click the [Solution] button for the error message.

   Click [Recheck the status] to reconfirm the storage destination resources.

3. Run recovery

   Restore the database cluster after WebAdmin detects an error.

   Refer to "8.2.1 Using WebAdmin" for details.

8.10.4.2 Using Server Commands

Follow the procedure below to recover the database.

1. Delete the data storage destination directory and the transaction log storage destination directory

   Save the data storage destination directory and the transaction log storage destination directory, and then delete them.

2. Execute recovery

   Use the pgx_rcvall command to recover the database cluster.

   Refer to "8.2.2 Using the pgx_rcvall Command" for details.

8.11 Actions in Response to Failure to Stop an Instance

If an instance fails to stop, refer to the system log and the server log, and determine the cause of the failure.

If the instance cannot stop despite taking action, perform the following operation to stop the instance.

There are two methods of recovery:

- 8.11.1 Using WebAdmin
- 8.11.2 Using Server Commands

8.11.1 Using WebAdmin

Click [Stop] in the [Monitor] window and select the Fast stop mode or the Immediate stop mode to stop the instance. Forcibly terminate the server process from WebAdmin if the instance cannot be stopped.

Refer to "2.1.1 Using WebAdmin" for information on the stop modes.

8.11.2 Using Server Commands

There are three methods:

- Stopping the Instance Using the Fast Mode

  If backup is in progress, then terminate it, roll back all executing transactions, forcibly close client connections, and then stop the instance.

- Stopping the Instance Using the Immediate Mode

  Forcibly terminate the instance immediately. A crash recovery is run when the instance is restarted.

- Forcibly Stopping the Server Process

  Reliably stops the server process when the other methods are unsuccessful.

8.11.2.1 Stopping the Instance Using the Fast Mode

Specify "-m fast" in the pg_ctl command to stop the instance.

If the instance fails to stop when you use this method, stop the instance as described in "8.11.2.2 Stopping the Instance Using the Immediate Mode" or "8.11.2.3 Forcibly Stopping the Server Process".

 Example

```
> pg_ctl stop -D /database/inst1 -m fast
```

8.11.2.2 Stopping the Instance Using the Immediate Mode

Specify "-m immediate " in the pg_ctl command to stop the instance.

If the instance fails to stop when you use this method, stop the instance as described in "8.11.2.3 Forcibly Stopping the Server Process".

 Example

```
> pg_ctl stop -D /database/inst1 -m immediate
```

8.11.2.3 Forcibly Stopping the Server Process

If both the Fast mode and the Immediate mode fail to stop the instance, use the kill command or the kill parameter of the pg_ctl command to forcibly stop the server process.

The procedure is as follows:

1. Execute the ps command

```
> ps axwfo user,pid,ppid,tty,command | grep postgres
fsepuser 19174 18027 pts/1                \_ grep postgres
```

```
fsepuser 20517     1 ?        /opt/fsepserver64/bin/postgres -D /database/inst1
fsepuser 20518 20517 ?         \_ postgres: logger process
fsepuser 20520 20517 ?         \_ postgres: checkpointer process
fsepuser 20521 20517 ?         \_ postgres: writer process
fsepuser 20522 20517 ?         \_ postgres: wal writer process
fsepuser 20523 20517 ?         \_ postgres: autovacuum launcher process
fsepuser 20524 20517 ?         \_ postgres: archiver process
fsepuser 20525 20517 ?         \_ postgres: stats collector process
```

The process ID (20517) indicates the server process.

2. Forcibly stop the server process

   As instance manager, forcibly stop the server process.

   Using the pg_ctl command

   ```
   > pg_ctl kill SIGQUIT 20517
   ```

   Using the kill command

   ```
   > kill -s SIGQUIT 20517
   ```

8.11.3 Errors Caused by Failure to Stop PL/extJava

If an error occurs when PL/extJava is stopped and the instance fails to stop, take the following corrective action:

- Check if error messages have been output to the following logs.

  Refer to "Messages Output by PL/extJava" in Messages for information on the output error messages, and investigate the cause of the failure.

  - System log
  - Enterprise Postgres server log
  - Container server log
  - Container Java VM log
  - Domain server log
  - Domain Java VM log

In some cases, processes may remain unduly, depending on the error that occurred when PL/extJava was stopped. Refer to "8.10.3 Errors Caused by Failure to Start PL/extJava" if this occurs, and end any processes that still exist.

If you cannot determine the cause of the error even after conducting the above investigation, use FJQSS to collect information, and then contact Fujitsu technical support.

8.12 Actions in Response to Error in a Distributed Transaction

If a system failure (such as server failure) occurs in an application that uses distributed transactions (such as .NET TransactionScope), then transactions may be changed to the in-doubt state. At that point, resources accessed by the transaction will be locked, and rendered unusable by other transactions.

The following describes how to check for in-doubt transactions, and how to resolve them.

How to check for in-doubt transactions

The following shows how to check for them:

If the server fails

1. An in-doubt transaction will have occurred if a message similar to the one below is output to the log when the server is restarted.

   Example

   ```
   LOG: Restoring prepared transaction 2103.
   ```

2. Refer to system view pg_prepared_xacts to obtain information about the prepared transaction.

   If the transaction identifier of the prepared transaction in the list (in the transaction column of pg_prepared_xacts) is the same as the identifier of the in-doubt transaction obtained from the log output when the server was restarted, then that row is the information about the in-doubt transaction.

   Example

   ```
   postgres=# select * from pg_prepared_xacts;
    transaction |   gid     |   prepared |  owner   | database
   -------------+-----------+------------+----------+----------
   2103 | 374cc221-f6dc-4b73-9d62-d4fec9b430cd | 2015-05-06 16:28:48.471+08 | postgres |
   postgres (1 row)
   ```

   Information about the in-doubt transaction is output to the row with the transaction ID 2103 in the transaction column.

If the client fails

If there are no clients connected and there is a prepared transaction in pg_prepared_xacts, then you can determine that the transaction is in the in-doubt state.

If at least one client is connected and there is a prepared transaction in pg_prepared_xacts, you cannot determine whether there is a transaction in the in-doubt state. In this case, use the following query to determine the in-doubt transaction from the acquired database name, user name, the time PREPARE TRANSACTION was executed, and the information about the table name accessed.

```
select gid,x.database,owner,prepared,l.relation::regclass as relation from pg_prepared_xacts x
left join pg_locks l on l.virtualtransaction = '-1/'||x.transaction and l.locktype='relation';
```

If it still cannot be determined from this information, wait a few moments and then check pg_prepared_xacts again.

If there is a transaction that has continued since the last time you checked, then it is likely that it is the one in the in-doubt state.

Point

As you can see from the explanations in this section, there is no one way to definitively determine in-doubt transactions.

Consider collecting other supplementary information (for example, logging on the client) or performing other operations (for example, allocating database users per job).

How to resolve in-doubt transactions

From the system view pg_prepared_xacts mentioned above, obtain the global transaction identifier (in the gid column of pg_prepared_xacts) for the in-doubt transaction, and issue either a ROLLBACK PREPARED statement or COMMIT PREPARED statement to resolve the in-doubt transaction.

Example

- Rolling back in-doubt transactions

```
postgres=# rollback prepared '374cc221-f6dc-4b73-9d62-d4fec9b430cd';
ROLLBACK PREPARED
```

- Committing in-doubt transactions

```
postgres=# commit prepared '374cc221-f6dc-4b73-9d62-d4fec9b430cd';
COMMIT PREPARED
```

8.13 I/O Errors Other than Disk Failure

Even if a disk is not defective, the same input-output error messages, as those generated when the disk is defective, may be output.

A few examples of such errors are given below. The appropriate action for each error is explained respectively.

- 8.13.1 Network Error with an External Disk
- 8.13.2 Errors Caused by Power Failure or Mounting Issues

8.13.1 Network Error with an External Disk

This is an error that occurs in the network path to/from an external disk.

Determine the cause of the error by checking the information in the system log and the server log, the disk access LED, network wiring, and network card status. Take appropriate action to remove the cause of the error, for example, replace problematic devices.

8.13.2 Errors Caused by Power Failure or Mounting Issues

These are errors that occur when the disk device is not turned on, automatic mounting of the disk was not set, or mounting was accidentally cancelled.

In this case, check the information in the system log and the server log, the disk access LED, and whether the disk is mounted correctly. If problems are detected, take appropriate action.

If mounting has been cancelled, it is possible that mounting was accidentally cancelled, or automatic mounting at the time of starting the operating system is not set. In this case, set the mounting to be performed automatically.

8.14 Operational Errors in PL/extJava Operations

If the pgx_jadmin command ends in an error, refer to the output error message, together with "Messages Output by PL/extJava" in Messages, and resolve the issue. If required, also refer to the PL/extJava log. Refer to "Appendix G PL/extJava Log Information" for information on the PL/extJava log.

The pgx_jadmin command runs in the PL/extJava environment, so any errors that might occur are almost always the same as those that occur when an instance is starting up in PL/extJava. Refer to "8.10.3 Errors Caused by Failure to Start PL/extJava" for information on typical examples.

8.15 Errors Related to Application Operations (during PL/extJava Operations)

8.15.1 Java Function Errors

If execution of a Java application fails, the Java function will return the error SQLSTATE 39000. Check the server log of Enterprise Postgres and then if required, check if any errors from IJServer12000 to IJServer12999 have been output to the logs below:

- System log
- Container server log
- Container Java VM log
- Domain server log
- Web server trace log

- Web server internal log

Some examples are provided below.

Abnormal termination of the Java EE DAS service

Even if the Java EE DAS service ends in an error, it will be restarted automatically, enabling jobs to continue. However, if the error occurs repeatedly, investigate its cause by referring to "Messages Output by PL/extJava" in Messages, using the messages with "IJServer" in the message number that are output to the log file below. Identify the cause and prevent the error from recurring.

- System log
- Domain server log

Abnormal termination of the PCMI service

The server instance(Java VM) monitored by the PCMI service and the Java EE DAS service will remain unduly. You must forcibly terminate these processes (refer to "8.10.3 Errors Caused by Failure to Start PL/extJava" for details), remove the cause of the error, and then restart the instance.

Java heap insufficiency

If an error message is output to the log below indicating that memory is insufficient and the Java application cannot be executed, then the heap area size and Perm area size of the server instance(Java VM) in the container must be modified.

- Container server log
- Container server instance(Java VM) log

Point

The following default values are set for the heap area and Perm area of the server instance(Java VM) in the container.

Type	Setting value
Server instance(Java VM) heap area size	512MB
Server instance(Java VM) Perm area size	192MB

Perform the following procedure to change the server instance(Java VM) heap area size and Perm area size in the container:

1. Use the pgx_jadmin command with the list-jvm-options subcommand to check the size of the heap area and Perm area in the current server instance(Java VM).

   #### Example

   ```
   $ pgx_jadmin list-jvm-options --dbname db01
   ```

   Refer to the output results of the corresponding item to be checked below for the heap area size and the Perm area size.

Option	Item to be checked
heap area size	-Xmx
Perm area size	-XX:MaxPermSize

   #### See

   Refer to "pgx_jadmin" in the Reference for information on the list-jvm-options subcommand.

2. Stop the instance

   Stop the instance . Refer to 2.1 Starting and Stopping an Instance" for information on how to stop an instance.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

3. Use the pgx_jadmin command with the modify-jvm-options subcommand to modify the configured size of the heap area and Perm area in the server instance(Java VM).

   Example

   To change the heap area size and the Perm area size of the server instance(Java VM) to "1024 (megabytes)" and "384 (megabytes)", respectively.

   ```
   $ pgx_jadmin modify-jvm-options --dbname db01 --heapsize 1024 --permsize 384
   ```

   See

   Refer to "pgx_jadmin" in the Reference for information on the modify-jvm-options subcommand.

4. Start the instance

   Start the instance. Refer to 2.1 Starting and Stopping an Instance" for information on how to start an instance.

   If the instance fails to start, refer to "8.10 Actions in Response to Instance Startup Failure"

Domain root disk failure

Refer to "8.10.3 Errors Caused by Failure to Start PL/extJava" for information on the corrective action to take when the disk on which the domain root is placed fails, or when domain root resources become corrupted due to incorrect operation.

8.15.2 No Response from Java Functions

Possible causes for Java functions not responding for an extended period of time are described below. Take the corresponding corrective action accordingly.

A Java application is attempting to update a record that has already been updated by the source calling the Java function

The source calling the Java function and the Java application are handled as separate transactions. Therefore, if the Java application attempts to update a record that has already been updated by the Java function, the lock processing will never be processed. This kind of situation can be checked using the pg_locks view (refer to "Viewing Locks" in "Monitoring Database Activity" under "Server Administration" in the PostgreSQL Documentation for details).

The Java application processing includes a process that is taking a long time

Use the jstack command to collect a full thread dump for the process ID listed in the file below, and check the content of the process being executed (refer to "F.1 Notes on Using jstack" for information on the jstack command).

```
domainRoot/domains/domain1/config/pid
```

Appendix A Parameters

This appendix describes the parameters to be set in the postgresql.conf file of Enterprise Postgres.

The postgresql.conf file is located in the data storage destination.

- core_directory (string)

  This parameter specifies the directory where the corefile is to be output. If this parameter is omitted, the data storage destination is used by default. This parameter can only be set when specified on starting an instance. It cannot be changed dynamically, while an instance is active.

- core_contents (string)

  This parameter specifies the contents to be included in the corefile.

  - full: Outputs all contents of the server process memory to the corefile.
  - none: Does not output a corefile.
  - minimum: Outputs only non-shared memory server processes to the corefile. This reduces the size of the corefile. However, in some cases, this file may not contain sufficient information for examining the factor that caused the corefile to be output.

  If this parameter is omitted, "minimum" is used by default. This parameter can only be set when specified on starting an instance. It cannot be changed dynamically, while an instance is active.

- keystore_location (string)

  This parameter specifies the directory that stores the keystore file. Specify a different location from other database clusters. This parameter can only be set when specified on starting an instance. It cannot be changed dynamically, while an instance is active.

- tablespace_encryption_algorithm (string)

  This parameter specifies the encryption algorithm for tablespaces that will be created. Valid values are AES128, AES256, and none. If you specify "none", encryption is not performed. The default value is "none". To perform encryption, it is recommended that you specify AES256. Only superusers can change this setting.

- backup_destination (string)

  This parameter specifies the absolute path of the directory where pgx_dmpall will store the backup data. Specify a different location from other database clusters. This parameter can only be set when specified on starting an instance. It cannot be changed dynamically, while an instance is active.

  Place this directory on a different disk from the data directory to be backed up and the tablespace directory. Ensure that users do not store arbitrary files in this directory, because the contents of this directory are managed by the database system.

- search_path (string)

  When using the SUBSTR function compatible with Oracle databases, set "oracle" and "pg_catalog" in the search_path parameter. You must specify "oracle" before "pg_catalog".

Example

```
search_path = '"$user", public, oracle, pg_catalog'
```

Information

  - The search_path feature specifies the priority of the schema search path. The SUBSTR function in Oracle database is defined in the oracle schema.
  - Refer to "Statement Behavior" under "Server Administration" in the PostgreSQL Documentation for information on search_path.

- track_waits (string)

  This parameter enables collection of statistics for pgx_stat_lwlock and pgx_stat_latch.

  - on: Enables collection of statistics.
  - off: Disables collection of statistics.

  If this parameter is omitted, "on" is assumed.

  Only superusers can change this setting.

- track_sql (string)

  This parameter enables collection of statistics for pgx_stat_sql.

  - on: Enables collection of statistics.
  - off: Disables collection of statistics.

  If this parameter is omitted, "on" is assumed.

  Only superusers can change this setting.

See

Refer to "Server Configuration" under "Server Administration" in the PostgreSQL Documentation for information on other postgresql.conf parameters.

Appendix B System Administration Functions

This appendix describes the system administration functions of Enterprise Postgres.

See

Refer to "System Administration Functions" under "The SQL Language" in the PostgreSQL Documentation for information on other system administration functions.

B.1 WAL Mirroring Control Functions

The following table lists the functions that can be used for backup and recovery based on WAL mirroring.

Table B.1 WAL mirroring control functions

Name	Return type	Description
pgx_pause_wal_multiplexing()	void	Stops WAL multiplexing
pgx_resume_wal_multiplexing()	void	Resumes WAL multiplexing
pgx_is_wal_multiplexing_paused()	boolean	Returns true if WAL multiplexing has stopped

If WAL multiplexing has not been configured, these functions return an error. Setting the backup_destination parameter in postgresql.conf configures WAL multiplexing.

Only superusers can execute these functions.

B.2 Transparent Data Encryption Control Functions

The following table lists the functions that can be used for transparent data encryption.

Table B.2 Transparent data encryption control functions

Name	Return type	Description
pgx_open_keystore(*passphrase*)	void	Opens the keystore
pgx_set_master_key(*passphrase*)	void	Sets the master encryption key
pgx_set_keystore_passphrase(*oldPassphrase*, *newPassphrase*)	void	Changes the keystore passphrase

The pgx_open_keystore function uses the specified passphrase to open the keystore. When the keystore is opened, the master encryption key is loaded into the database server memory. In this way, you can access the encrypted data and create encrypted tablespaces. If the keystore is already open, this function returns an error.

Only superusers can execute this function. Also, this function cannot be executed within a transaction block.

The pgx_set_master_key function generates a master encryption key and stores it in the keystore. If the keystore does not exist, this function creates a keystore. If the keystore already exists, this function modifies the master encryption key. If the keystore has not been opened, this function opens it.

The passphrase is a string of 8 to 200 bytes.

Only superusers can execute this function. Also, this function cannot be executed within a transaction block. Processing is not affected by whether the keystore is open.

The pgx_set_keystore_passphrase function changes the keystore passphrase. Specify the current passphrase in *oldPassphrase*, and a new passphrase in *newPassphrase*.

The passphrase is a string of 8 to 200 bytes.

Only superusers can execute this function. Also, this function cannot be executed within a transaction block. Processing is not affected by whether the keystore is open.

Appendix C System Views

This appendix describes how to use the system views in Enterprise Postgres.

See

Refer to "System Views" under "Internals" in the PostgreSQL Documentation for information on other system views.

C.1 pgx_tablespaces

The pgx_tablespaces catalog provides information related to the encryption of tablespaces.

Name	Type	References	Description
spctablespace	oid	pg_tablespace.oid	Tablespace OID
spcencalgo	text		Tablespace encryption algorithm

The spcencalgo string displays one of the following values:

- none: Tablespace is not encrypted
- AES128: AES with key length of 128 bits
- AES256: AES with key length of 256 bits

C.2 pgx_stat_lwlock

The pgx_stat_lwlock view displays statistics related to lightweight locks, with each type of content displayed on a separate line.

Table C.1 pgx_stat_lwlock view

Column	Type	Description
lwlock_name	name	Name of the lightweight lock
total_waits	bigint	Number of waits caused by the lightweight lock
total_wait_time	double precision	Number of milliseconds spent in waits caused by the lightweight lock
stats_reset	timestamp with timezone	Last time at which this statistics was reset

C.3 pgx_stat_latch

The pgx_stat_latch view displays statistics related to latches, with each type of wait information within Enterprise Postgres displayed on a separate line.

Table C.2 pgx_stat_latch view

Column	Type	Description
latch_name	name	Name of the latch
total_waits	bigint	Number of waits caused a wait
total_wait_time	double precision	Number of milliseconds spent in waits caused by the latch
stats_reset	timestamp with timezone	Last time at which this statistic was reset

C.4 pgx_stat_walwriter

The pgx_stat_walwriter view display statistics related to WAL writing, in a single line.

Table C.3 pgx_stat_walwriter view

Column	Type	Description
dirty_writes	bigint	Number of times old WAL buffers were written to the disk because the WAL buffer was full when WAL records were added
writes	bigint	Number of WAL writes
write_blocks	bigint	Number of WAL write blocks
total_write_time	double precision	Number of milliseconds spent on WAL writing
stats_reset	timestamp with timezone	Last time at which this statistic was reset

C.5 pgx_stat_sql

The pgx_stat_sql view displays statistics related to SQL statement executions, with each type of SQL statement displayed on a separate line.

Table C.4 pgx_stat_sql view

Column	Type	Description
selects	bigint	Number of SELECT statements executed
inserts	bigint	Number of INSERT statements executed
deletes	bigint	Number of DELETE statements executed
updates	bigint	Number of UPDATE statements executed
declares	bigint	Number of DECLARE statements executed (number of cursor OPENs)
fetches	bigint	Number of FETCH statements executed
checkpoints	bigint	Number of CHECKPOINT statements executed
clusters	bigint	Number of CLUSTER statements executed
copies	bigint	Number of COPY statements executed
reindexes	bigint	Number of REINDEX statements executed
truncates	bigint	Number of TRUNCATE statements executed
locks	bigint	Number of times a lock occurred
stats_reset	timestamp with timezone	Last time at which this statistic was reset

Appendix D Activating and Stopping the Web Server Feature of WebAdmin

To use WebAdmin for creating and managing a Enterprise Postgres instance on a server where Enterprise Postgres is installed, you must first activate the Web server feature of WebAdmin.

This appendix describes how to activate and stop the Web server feature of WebAdmin.

D.1 Activating the Web Server Feature of WebAdmin

Follow the procedure below to activate the Web server feature of WebAdmin.

1. Change to superuser

   Acquire superuser privileges on the system.

   Example

   ```
   $ su -
   Password:******
   ```

2. Activate the Web server feature of WebAdmin

   Execute the WebAdminStart command to activate the Web server feature of WebAdmin.

   Example

   If Enterprise Postgres is installed in /opt/fsepserver64:

   ```
   # cd /opt/fsepserver64/gui/sbin
   # ./WebAdminStart
   ```

D.2 Stopping the Web Server Feature of WebAdmin

This section describes how to stop the Web server feature of WebAdmin.

Follow the procedure below to stop the Web server feature of WebAdmin.

1. Change to superuser

   Acquire superuser privileges on the system.

   Example

   ```
   $ su -
   Password:******
   ```

2. Stop the Web server feature of WebAdmin

   Execute the WebAdminStop command to stop the Web server feature of WebAdmin.

   Example

   If Enterprise Postgres is installed in /opt/fsepserver64:

   ```
   # cd /opt/fsepserver64/gui/sbin
   # ./WebAdminStop
   ```

Appendix E Collecting Failure Investigation Data

If the cause of an error that occurs while building the environment or during operations is unclear, data must be collected for initial investigation.

This appendix describes how to collect data for initial investigation.

Use FJQSS (Information Collection Tool) to collect data for initial investigation.

See

Refer to the FJQSS manual for information on how to use FJQSS.

Note

When using FJQSS to collect data for initial investigation, you must set the following environment variables:

- Environment variables required for using Enterprise Postgres

  Refer to "Setup" in the Install and Setup Guide for Server for details.

- PGDATA

  Set the data storage destination.

- PGPORT

  Set the instance port number. This does not need to be set if the default port number (27500) has not been changed.

- PGUSER

  Set the database superuser.
  Set the database superuser so that client authentication is possible.
  FJQSS establishes a TCP/IP connection with the template1 database and collects data from the database.

- FSEP_HOME

  Set the Enterprise Postgres installation directory.

In addition, when using database multiplexing, set the following environment variables:

- MCCONTROLDIR

  Refer to "Mirroring Controller Resources" in the Cluster Operation Guide for information on the Mirroring Controller management directory.

Appendix F Notes on PL/extJava

This section provides some points to note when using PL/extJava.

F.1 Notes on Using jstack

Note the following when using the jstack tool provided in JDK 7 for troubleshooting:

- You cannot use the -m option.

See

Information about the jstack tool is provided in JDK documentation. JDK documentation installed on this product is available from the following URL:

http://docs.oracle.com/javase/7/docs/

Appendix G PL/extJava Log Information

Errors and warnings in PL/extJava environments are output to log files.

This information is helpful for detecting whether an error has occurred, and if so, what was the cause.

G.1 Domain

This section explains the logs output by domains.

The logs described below are output by domains that control Java EE DAS services and containers.

G.1.1 Server Log

Output destination

The server log is output to the following file:

```
domainRoot/domains/domain1/logs/server.log
```

Output details

- Number of generations

  1

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the maximum log size If the maximum log size is exceeded, old information will be saved to a file with the original name followed by rotation datetime.

  [Example]

```
server.log_2012-01-07T15-23-30
```

Output example

```
 [#|2015-03-18T18:20:16.653+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1010: Entering Security
Startup Service|#]

[#|2015-03-18T18:20:16.653+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1143: Loading policy
provider com.sun.enterprise.security.provider.PolicyWrapper.|#]

[#|2015-03-18T18:20:17.230+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1115: Realm [admin-realm] of
classtype [com.sun.enterprise.security.auth.realm.file.FileRealm] successfully created.|#]

[#|2015-03-18T18:20:17.261+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1115: Realm [file] of
classtype [com.sun.enterprise.security.auth.realm.file.FileRealm] successfully created.|#]

[#|2015-03-18T18:20:17.277+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1115: Realm [certificate] of
classtype [com.sun.enterprise.security.auth.realm.certificate.CertificateRealm] successfully
created.|#]
```

G.1.2 Java VM Log

Output destination

The Java VM log is output to the following file:

```
domainRoot/pcmi/isje6/logs/server/console.log
```

Output details

- Number of generations

  1

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the maximum log size is exceeded, old information will be saved to a file with the original name followed by rotation datetime.

  [Example]

```
console.log_2015_05_14-15_56_51
```

Output example

```
--------------------------------------------------------------------------------
                            12/05/2015  10:53:51
--------------------------------------------------------------------------------
/opt/fsepserver64/java/jdk7/bin/java
-XX:+PrintVMTerminatedMessage
-XX:InsufficientMemoryHandler=pcmiJavaVM-rhel6-amd64
-Xbootclasspath/a:/opt/fsepserver64/java/jdk7/jre/lib/fmoni.jar
-Xrunfmoni:detail=y
 :
 :
 :
Launching GlassFish on Felix platform
Completed shutdown of Log manager service
Completed shutdown of GlassFish runtime
#### JavaVM terminated: Java HotSpot(TM) 64-Bit Server VM (23.5.02_FUJITSU_MODIFIED-B04_NPTL mixed
mode), [pid=26436] TimeMillis=1399860897252 Time=Mon May 12 11:14:57 2015
```

G.2 Container

This section explains the logs output by containers.

The logs described below are output by applications that run on the server instance (JavaVM).

G.2.1 Server Log

Output destination

The server log is output to the following file:

```
domainRoot/nodes/localhost-domain1/ serverInstance(JavaVM)Name /logs/server.log
```

Output details

- Number of generations

  1

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the maximum log size If the maximum log size is exceeded, old information will be saved to a file with the original name followed by rotation datetime.

  [Example]

```
server.log_2012-01-07T15-23-30
```

Output example

```
[#|2015-03-18T18:20:16.653+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1010: Entering Security
Startup Service|#]

[#|2015-03-18T18:20:16.653+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1143: Loading policy
provider com.sun.enterprise.security.provider.PolicyWrapper.|#]

[#|2015-03-18T18:20:17.230+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1115: Realm [admin-realm] of
classtype [com.sun.enterprise.security.auth.realm.file.FileRealm] successfully created.|#]

[#|2015-03-18T18:20:17.261+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1115: Realm [file] of
classtype [com.sun.enterprise.security.auth.realm.file.FileRealm] successfully created.|#]

[#|2015-03-18T18:20:17.277+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1115: Realm [certificate] of
classtype [com.sun.enterprise.security.auth.realm.certificate.CertificateRealm] successfully
created.|#]
```

Message details

The following explains the messages output in PL/extJava environments.

- Output format

  Messages are output in the following format:

```
[#|yyyy-MM-ddTHH:mm:ss.msTZD|logLevel|||_ThreadID=IDthreadId;_ThreadName=threadName;|
messageBody|#]
```

  - *logLevel*

    Fixed to "INFO".

  - *messageBody*

    Text output by the PL/extJava environment.

Note the following about the log content:

- If a user Java application detects an exception, only the class name of the exception is output.
- For security reasons, the exception stack trace and exception message are not output.

- Output format (message body)

  The message body is output in the following format:

```
logLevel: fsep-plextjava: [resultCode] message [backend_process=pid]
```

  - *logLevel*

    Fixed to " ERROR".

Log level	Output content
ERROR	Error information when an error occurs

  - *resultCode*

    The following return values are returned:

Return value	Explanation
0	Returned successfully.
1001	An internal error was detected. (protocol rule violation) Contact Fujitsu technical support.
2001	An internal error was detected. (protocol rule violation) Contact Fujitsu technical support.
2002	The Java application to be run cannot be found. Check if the jar file of the Java application has been registered, the FUNCTION specification contains an error, or the created Java application contains an error. [Considerations] - Was the "*packageName.className.methodName*" specified during registration of FUNCTION correct? - Does the "(Java function) arguments" specified during registration of FUNCTION match the type and number of arguments of the created Java application? - Does the return value type specified during registration of FUNCTION match the type of the return values for methods of the created Java application? - Have the methods of the created Java application been created as static methods (static) with the access modifier "public"?
3001	The user does not have privileges to run the Java application. Check the privileges for running Java applications. [Considerations] - Has the access modifier of the class of the created Java application been created as "public"?
3002	An exception occurred while the Java application was running. Check the database status or the processing content of the Java application.
9001	An internal error was detected. (environment error) Contact Fujitsu technical support.

  - backend_process

    PID of the Enterprise Postgres (backend) process.

G.2.2 Java VM Log

Output destination

The Java VM log is output to the following file: name of the server instance (Java VM)

```
domainRoot/pcmi/isje6/logs/serverInstance(JavaVM)Name/console.log
```

Output details

- Number of generations

  1

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the maximum log size If the maximum log size is exceeded, old information will be saved to a file with the original name followed by rotation datetime.

  [Example]

  ```
  console.log_2015_05_14-15_56_51
  ```

Output example

```
--------------------------------------------------------------------------------
                             12/05/2015  10:53:51
--------------------------------------------------------------------------------
/opt/fsepserver64/java/jdk7/bin/java
-XX:+PrintVMTerminatedMessage
-XX:InsufficientMemoryHandler=pcmiJavaVM-rhel6-amd64
-Xbootclasspath/a:/opt/fsepserver64/java/jdk7/jre/lib/fmoni.jar
-Xrunfmoni:detail=y
 :
 :
 :
Launching GlassFish on Felix platform
Completed shutdown of Log manager service
Completed shutdown of GlassFish runtime
#### JavaVM terminated: Java HotSpot(TM) 64-Bit Server VM (23.5.02_FUJITSU_MODIFIED-B04_NPTL mixed
mode), [pid=26436] TimeMillis=1399860897252 Time=Mon May 12 11:14:57 2015
```

G.2.3 HTTP Access Log

Output destination

The HTTP access log is output to the following file:

```
domainRoot/nodes/localhost-domain1/serverInstance(JavaVM)Name/logs/access/server_access_log.txt
```

Output details

- Number of generations

  5

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  After one day, old information will be saved in a file that has the file name format "server_access_log.YYYY_MM_DD-hh_mm_ss.txt".
  If the rotation is performed within one second since the previous rotation, the file name format will be "server_access_log.YYYY_MM_DD-hh_mm_ss.*serialNumber*.txt".

Output example

```
"127.0.0.1" "-" "-" "14/May/2015:09:07:53 +0900" "POST /db01/callstored HTTP/1.1" "200" "1"
"127.0.0.1" "-" "4207" "ThreadID=68" "ThreadName=http-thread-pool-27532(1)" "127.0.0.1"
```

G.2.4 HTTP Trace Log

Output destination

The HTTP trace log is output to the following file:

```
domainRoot/nodes/localhost-domain1/serverInstance(JavaVM)Name/logs/http/trace.log
```

Output details

- Number of generations

  10

- Maximum log size

  10,485,760 bytes

- Length of Record

  Variable

- Rotation conditions

  If the maximum log size If the maximum log size is exceeded, old information will be saved to a file with the original name followed by rotation datetime, that is, "trace.log_YYYY_MM_DD-hh_mm_ss".
  If the rotation is performed within one second since the previous rotation, the file name format will be "trace.log_YYYY_MM_DD-hh_mm_ss.*serialNumber*".

Output example

```
"15/May/2015:11:21:31.608" "21(Grizzly-kernel-thread(1))" "conn" "127.0.0.1:60073"
"15/May/2015:11:21:31.609" "21(Grizzly-kernel-thread(1))" "qin" "127.0.0.1:60073"
"15/May/2015:11:21:31.609" "41(admin-thread-pool-27521(2))" "qout" "127.0.0.1:60073"
"15/May/2015:11:21:31.610" "41(admin-thread-pool-27521(2))" "recv" "GET /__asadmin/get?Xhelp=true
HTTP/1.1"
"15/May/2015:11:21:31.616" "41(admin-thread-pool-27521(2))" "send" "200" "127.0.0.1:60073"
"15/May/2015:11:21:31.626" "21(Grizzly-kernel-thread(1))" "k-wt" "127.0.0.1:60073"
"15/May/2015:11:21:31.704" "21(Grizzly-kernel-thread(1))" "qin" "127.0.0.1:60073"
"15/May/2015:11:21:31.704" "48(admin-thread-pool-27521(4))" "qout" "127.0.0.1:60073"
"15/May/2015:11:21:31.704" "48(admin-thread-pool-27521(4))" "recv" "GET /__asadmin/get?
DEFAULT=inst1.http-service.isjee-trace-log.max-history-files HTTP/1.1"
"15/May/2015:11:21:31.707" "48(admin-thread-pool-27521(4))" "send" "200" "127.0.0.1:60073"
"15/May/2015:11:21:31.711" "21(Grizzly-kernel-thread(1))" "k-wt" "127.0.0.1:60073"
"15/May/2015:11:21:31.758" "21(Grizzly-kernel-thread(1))" "qin" "127.0.0.1:60073"
"15/May/2015:11:21:31.758" "37(admin-thread-pool-27521(1))" "qout" "127.0.0.1:60073"
"15/May/2015:11:21:31.758" "21(Grizzly-kernel-thread(1))" "disc" "127.0.0.1:60073"
```

G.3 Web Server

This section explains the logs output by Web servers.

The logs described below are output by the web server that controls communication between the database and the server instance (JavaVM).

G.3.1 Error Log

Web server error information is output to an error log.

Output destination

The error log is output to the following file:

```
EnterprisePostgresInstallDir/java/ahs22/plextjava/domain1/logs/errorlog
```

Output details

- Number of generations

  5

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the log size is exceeded, old information will be saved to a file with the name "errorlog.N".

  N is a consecutive serial number, assigned in order of the newness of files - for the first file it is "0", and for the *n*th file it is "(N-1)".

- Output content

An error log is output every time an error occurs in a Web server.

The format of the log output as an error log is shown below.

```
[dateTime] [logLevel] [clientIpAddress] (errorNumber) errorDescription: messageBody
```

Output items

The following explains the output items in the output format.

- *dateTime*

  Datetime of error
  Output in the format "[*dayOfTheWeek month day hour*:*minute*:*second year*]".

- *logLevel*

Log level	Error log multiplicity	Output at initial setup
emerg	Error generated during an emergency	Y
alert	Error that prevents operations if not corrected	Y
crit	Error that should be handled immediately	Y
error	Minor error that can be ignored	Y
warn	Warning that can be ignored	Y
notice	Notification that can occur at any time and should be noted down	Y

Log level	Error log multiplicity	Output at initial setup
info	Notification other than notice Output when an environment definition is edited.	N
debug	Log during module development and debugging Output when an environment definition is edited.	N

- *clientIpAddress*

  IP address of the server, such as a client or proxy server
  Output when an error occurs during Web server access from a client.
  This item may be omitted.

- *errorNumber*

  Operating system error number
  This item may be omitted.

- *errorDescription*

  Description of the error number
  This item may be omitted.

- *messageBody*

  Body of the message

Output example

```
[Tue May 13 18:00:02 2015] [notice] Apache/2.2.22 (Unix) configured -- resuming normal operations
[Tue May 13 18:00:32 2015] [error] [client 127.0.0.1] File does not exist: /opt/fsepserver64/java/
ahs22/domain1/htdocs/db01
```

G.3.2 Trace Log

Web server input/output information is output to a trace log.

Output destination

The trace log is output to the following file:

```
EnterprisePostgresInstallDir/java/ahs22/plextjava/domain1/logs/tracelog
```

Output details

- Number of generations

  5

- Maximum log size

  2,097,152 bytes

- Length of Record

  Variable

- Rotation conditions

  If the log size is exceeded, old information will be saved to a file with the name "tracelog.N".

  N is a consecutive serial number, assigned in order of the newness of files - for the first file it is "0", and for the *n*th file it is "(N-1)".

- Output content

  The trace log is output at the times shown below for each item of trace information.

- Web client input/output information
  - When a TCP connection is established
  - When a TCP connection is closed
  - When an HTTP request is received
  - When an HTTP response is sent
- Plug-in module input/output information
  - When the plug-in module response processing function is called
  - When the plug-in module response processing function is returned

Output format

The format of the log output as a trace log is shown below.

```
[dateTime][processId][threadId]detailedEventInformation
```

Output items

The following explains the output items in the output format.

- *dateTime*

  Datetime of trace information output
  The datetime is output in the format "[*day*/*month*/*year*:*hour*:*minute*:*second*.*millisecond*]".

- *processId*

  Process ID of the communication process

- *threadId*

  Thread ID of the communication thread

- *detailedEventInformation*

  The following table explains the trace content formats for each output trigger.

Output trigger	Event	Detailed event information
When a TCP connection is established	conn	*IpAddrOfClient*:*portNum* =>*IpAddrOfTargetWebServer*:*PortNum* I
When a TCP connection is closed	disc	
When an HTTP request is received	recv	Content of the request line This portion contains the escaped request.
When an HTTP response is sent	send	Status code
When the plug-in module response processing function is called (*1)	call	Module source name (*2)
When the plug-in module response processing function is returned (*1)	rtn	Module source name (return code) (*2) The following are output as return codes: - When response processing in this module is not executed: -1 - When response processing in this module is executed: Other than -1

*1: The output target plug-in modules are those not provided in HTTP Server 2.2.

*2: This is the source file name when the plug-in module is compiled. The module source name of the Web server connector provided in the Servlet service is output as "mod_jk2.c".

Information

The HTTP Keep-Alive feature connection retention time can be checked from the trace log. For the output trace log, the datetime of disc (event when a TCP connection is closed) and send (event when an HTTP response is sent) is checked, and calculated according to the following formula:

```
httpKeepAliveConnectionRetentionTime= discDatetime - sendDatetime
```

Note: When a timeout occurs for a request from the web server to break the TCP connection to the client, if the disconnection notice from the client cannot be received within two seconds, the connection retention time may be two seconds longer than the setting in the KeepAliveTimeout directive..

Output example

```
[15/May/2015:11:34:34.550][5144][5157]conn 127.0.0.1:44217=>127.0.0.1:27530
[15/May/2015:11:34:34.550][5144][5157]recv "POST /db01/callstored HTTP/1.1"
[15/May/2015:11:34:34.551][5144][5157]call mod_jk2.c
[15/May/2015:11:34:34.576][5144][5157]rtn  mod_jk2.c(0)
[15/May/2015:11:34:34.576][5144][5157]send 400
[15/May/2015:11:34:34.591][5144][5157]disc
```

G.3.3 Access Log

In HTTP Server 2.2, the access status from the Web browser is output to an access log.

Output destination

The access log is output to the following file:

```
EnterprisePostgresInstallDir/java/ahs22/plextjava/domain1/logs/accesslog
```

Output details

- Number of generations

  5

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the log size is exceeded, old information will be saved to a file with the name "accesslog.N".

  N is a serial number, assigned consecutively starting at 0. The serial numbers are assigned in order of the newness of files, with the first file having the serial number "0" and the *n*th file having the serial number "(N-1)".

- Output content

  The access log is output when the Web server receives a request from a client and sends a response to the client.

Output format

The format of logs output in the default value format for the access log (ahs-analysis) is shown below. Items not specified during access are output as a hyphen "-".

```
hostName userNameIdentifier userName dateTime "request" statusCode dataTransferVolume
ipAddressOfWebServer:portNumber hostHeader processId processingTime requestId
```

Output items

The following table explains the output items in the output format.

Output item	Output content
hostName	IP address or host name of the client .
userNameIdentifier	Personal user information returned from the client.
userName	User name sent from the client.
dateTime	Datetime at which the request was received from the client. Output in the format "[*day*/*month*/*year*:*hour*:*minute*:*second timeDifferenceFromGMT*]".
request	Request content from the client.
statusCode	Code returned to the client.
dataTransferVolume	Amount of data transferred to the client.
ipAddressOfWebServer:*portNumber*	IP address and port number of the Web server that received the request.
hostHeader	Host header content sent from the client.
processId	Process ID of the process that processed the request.
processingTime	Time from when the request was received until processing was completed. Output in the format "*microsecond*".
requestId	Unique ID granted per request

Output example

```
127.0.0.1 - - [15/May/2015:11:32:17 +0900] "POST /db01/callstored HTTP/1.1" 200 3 127.0.0.1:80
127.0.0.1 5143 26901 -
```

G.3.4 Internal Log

Web server connector error information is output to a log.

Output destination

Error logs are output to the following file:

```
EnterprisePostgresInstallDir/java/wsc/logs/jk2/httpd_domain1.conf/jk2.log
```

Output details

- Number of generations

  2

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the specified log size or the specified time is exceeded, old information will be saved to a file with the name "jk2_YY.MM.DD_hh.mm.ss.log".

Output example

```
[15/05/2015 09:25:08:947 +0900] ( info) IJServer12047: Web Server Connector running. conf="/opt/
fsepserver64/java/wsc/conf/jk2/httpd_domain1.conf/workers2.properties" pid=8621 tid=4294967295
[15/05/2015 09:36:03:596 +0900] ( info) IJServer12048: Web Server Connector stops running. conf="/opt/
fsepserver64/java/wsc/conf/jk2/httpd_domain1.conf/workers2.properties" pid=8621 tid=4294967295
```

Index

[W]

FUJITSU

FUJITSU Enterprise Postgres 9.4

Operation Guide

Windows

JB1WS-1222-01ENZ0(00)
September 2015

Preface

Purpose of this document

The Enterprise Postgres database system extends the PostgreSQL features and runs on the Windows platform.

This document is the Enterprise Postgres Operation Guide.

Intended readers

This document is intended for those who install and operate Enterprise Postgres.

Readers of this document are assumed to have general knowledge of:

- PostgreSQL
- SQL
- Windows

Structure of this document

This document is structured as follows:

Appendix F Notes on PL/extJava

Describes the note on PL/extJava.

Appendix G PL/extJava Log Information

Describes the PL/extJava log information.

Export restrictions

Exportation/release of this document may require necessary procedures in accordance with the regulations of your resident country and/or US export control laws.

Issue date and version

```
First edition: September 2015
```

Copyright

Contents

Chapter 1 Operating Enterprise Postgres

This chapter describes how to operate Enterprise Postgres.

1.1 Operating Methods

There are two methods of managing Enterprise Postgres operations:

- Operation management using GUI tools
- Operation management using commands

See

Before performing switchover or failover operation using database multiplexing, refer to "Database Multiplexing Mode" in the Cluster Operation Guide.

Operation management using GUI tools

This involves managing operations using the WebAdmin and pgAdmin GUI tools.

- Management using WebAdmin

  This removes the requirement for complex environment settings and operational design for backup and recovery that is usually required for running a database. It enables you to easily and reliably monitor the state of the database, back up the database, and restore it even if you do not have expert knowledge of databases.

- Management using pgAdmin

  When developing applications and maintaining the database, you can use pgAdmin to perform simple operations on database objects, such as:

  - Rebuild indexes and update statistics
  - Create, delete, and update database objects

  In addition, from pgAdmin of Enterprise Postgres, you can use the expanded features provided by Enterprise Postgres on the PostgreSQL SQL commands.

  See

  Refer to pgAdmin Help for information on the expanded features of pgAdmin provided by Enterprise Postgres.

Operation management using commands

You can use commands for configuring and operating the database and managing operations. However, note that if you start managing operations using commands, you cannot switch to WebAdmin-based operation management.

Note

You cannot combine WebAdmin and server commands to perform the following operations:

- Use WebAdmin to operate an instance created using the initdb command
- Use commands to operate an instance created using WebAdmin
- Use WebAdmin to recover a database backed up using commands

For instances created with WebAdmin, however, backup can be obtained with the pgx_dmpall command. Also, WebAdmin can perform recovery by using the backup obtained with the pgx_dmpall command.

- You can perform backup and restoration in pgAdmin, but the backup data obtained with WebAdmin and pgx_dmpall is not compatible with the backup data obtained with pgAdmin.
- Refer to pgAdmin Help for other notes on pgAdmin.

Features used in each phase

The following table lists the features used in each phase for GUI-based operations and command-based operations.

Operation		GUI-based operation	Command-based operation
Setup	Instance creation	WebAdmin	initdb command
	Modification of the configuration file	WebAdmin	Directly edit the configuration file
Instance start		WebAdmin	OS-provided net command or sc command
Database creation		pgAdmin	Specify using the DDL statement, and define using psql and applications
Database backup		WebAdmin pgx_dmpall command	pgx_dmpall command
Monitoring	Database failure	WebAdmin(*1)	Messages output to the event log (*1)
	Disk space	WebAdmin (*1) (*2)	OS-provided fsutil command (check available capacity) and dir command (check used capacity)
	Connection status	pgAdmin	psql command (*3)
Database recovery		WebAdmin	pgx_rcvall command

*1: Operations can be monitored using operation management middleware (such as Systemwalker Centric Manager).

*2: A warning is displayed when disk usage reaches 80%.

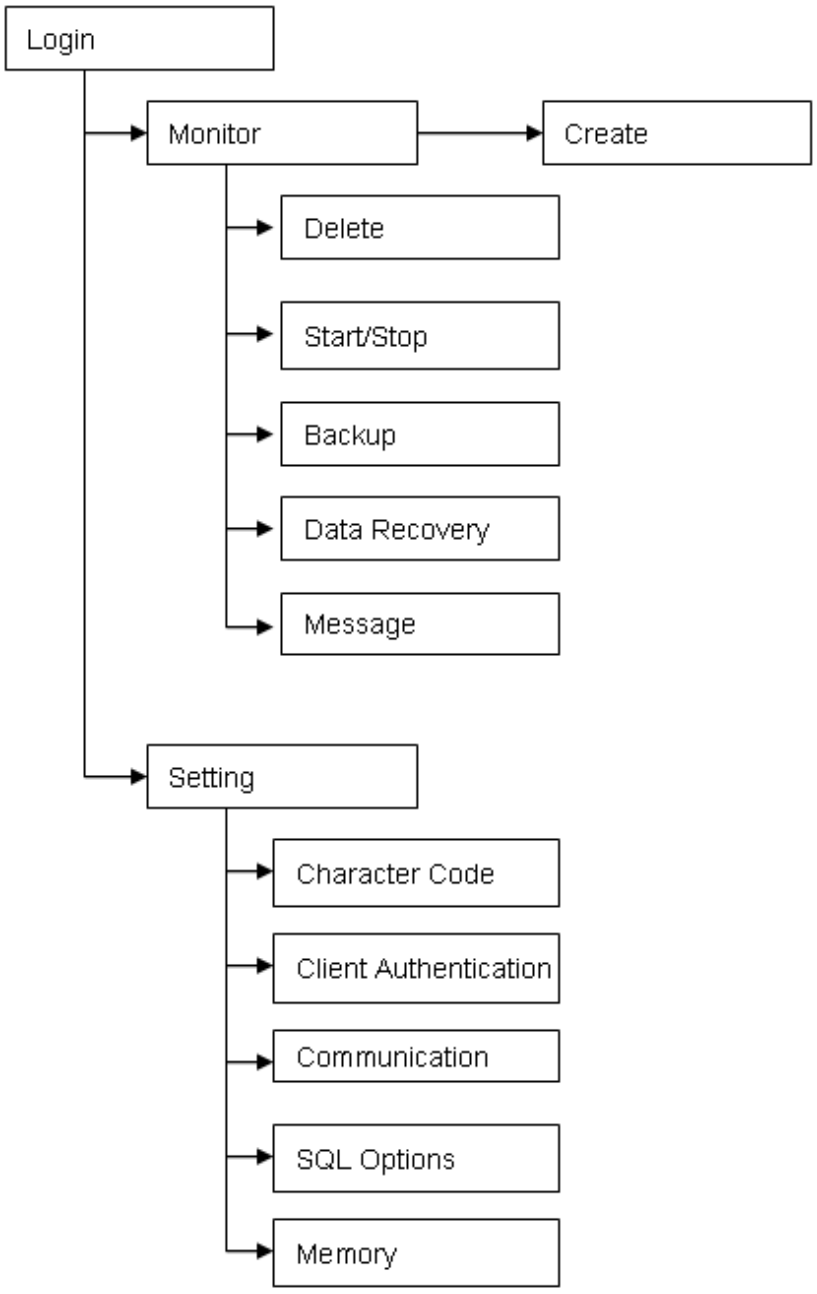
*3: This command searches for pg_stat_activity in the standard statistics views and monitors the state.

1.2 Activating WebAdmin

This section describes how to activate and log in to WebAdmin.

1.2.1 Flow of WebAdmin

The figure below shows the flow of WebAdmin GUI windows.



Monitor menu

Using this menu, you can operate the following instances and display their states:

- [Create]: Creates a database cluster and instance
- [Delete]: Deletes a database cluster and instance
- [Start/Stop]: Starts or stops an instance
- [Backup]: Performs back up of a database cluster
- [Data Recovery]: Recovers a database cluster
- [Message]: Displays messages about operations performed using WebAdmin, and about errors that are detected

See

Refer to the following for information on the functionality available from the [Monitor] menu:

- Creation or deletion: "Creating an Instance" in the Installation and Setup Guide for Server
- Starting and stopping: "2.1.1 Using WebAdmin"
- Backup: "3.2.1 Using WebAdmin"
- Data recovery: "8.3.1 Using WebAdmin"

Setting menu

Using this menu, you can set the definition information for the following instances:

- [Character Code]: Sets the character set and locale
- [Client Authentication]: Sets the authentication information to be used when a client connects to an instance
- [Communication]: Sets the communication definition for applications and instances
- [SQL Options]: Sets the definition to be used when executing an SQL statement
- [Memory]: Sets the memory to be used

See

Refer to "Changing the settings" in the Installation and Setup Guide for Server for information on the [Setting] menu.

1.2.2 Logging in to WebAdmin

This section describes how to log in to WebAdmin.

User environment

The following browser is required for using WebAdmin:

- Internet Explorer 8.0 or later

Activation URL for WebAdmin

In the browser address bar, type the activation URL of the WebAdmin window in the following format:

```
http://hostNameOrIpAddress:portNumber/
```

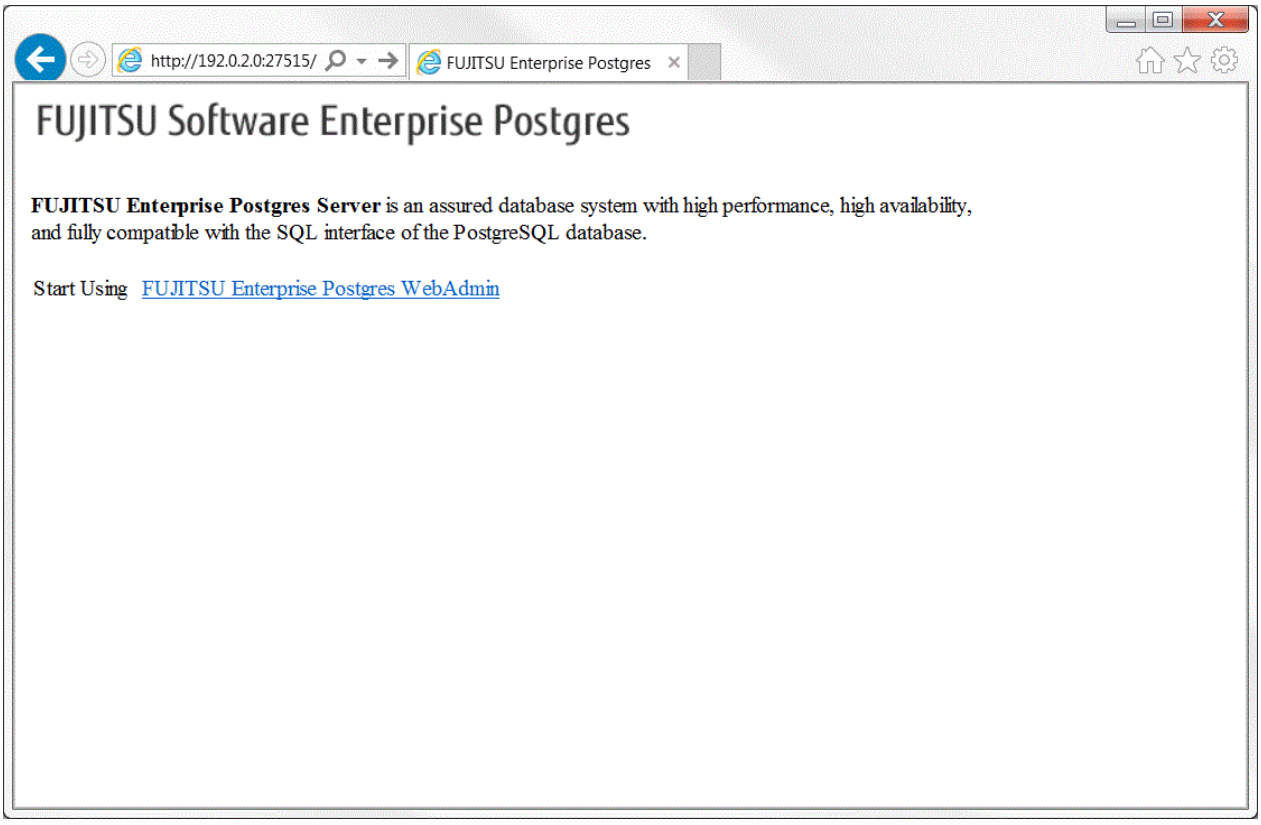
- *hostNameOrIpAddress*: The host name or IP address of the server where Enterprise Postgres is installed.
- *portNumber*: The port number of WebAdmin. The default port number is 27515.

Example

For a server with IP address "192.0.2.0" and port number "27515"

```
http://192.0.2.0:27515/
```

The activation URL window shown below is displayed.



Point

- You must activate the Web server feature of WebAdmin before using WebAdmin.
- Refer to "Appendix D Activating and Stopping the Web Server Feature of WebAdmin" for information on how to activate the Web server feature of WebAdmin.

Log in to WebAdmin

Click [FUJITSU Enterprise Postgres WebAdmin] in the activation URL window to activate WebAdmin and display the [Log in] window. You can log in to WebAdmin using the [Log in] window.

To log in, specify the following values:

- [User ID]: User ID (OS user account) of the instance administrator
- [Password]: Password corresponding to the user ID

Point

Use the OS user account as the user ID of the instance administrator. Refer to "Creating an Instance Administrator" in the Installation and Setup Guide for Server for details.

1.3 Starting pgAdmin

This section describes how to start pgAdmin, how to add an instance required for managing a database, and how to connect to and disconnect from the instance.

You can use pgAdmin on the Windows client.

1.3.1 Starting pgAdmin

This section explains how to start pgAdmin if you are using it from the product "Enterprise Postgres Client (*AA*bit) *x.y.z*" (where *AA* is "32" or "64", *x.y.z* is the version number(*x.y* SP*z*)).

Windows(R) 8 or Windows Server(R) 2012

From the [Start] screen, start [pgAdmin III(*AA*bit)(*x.y.z*)].

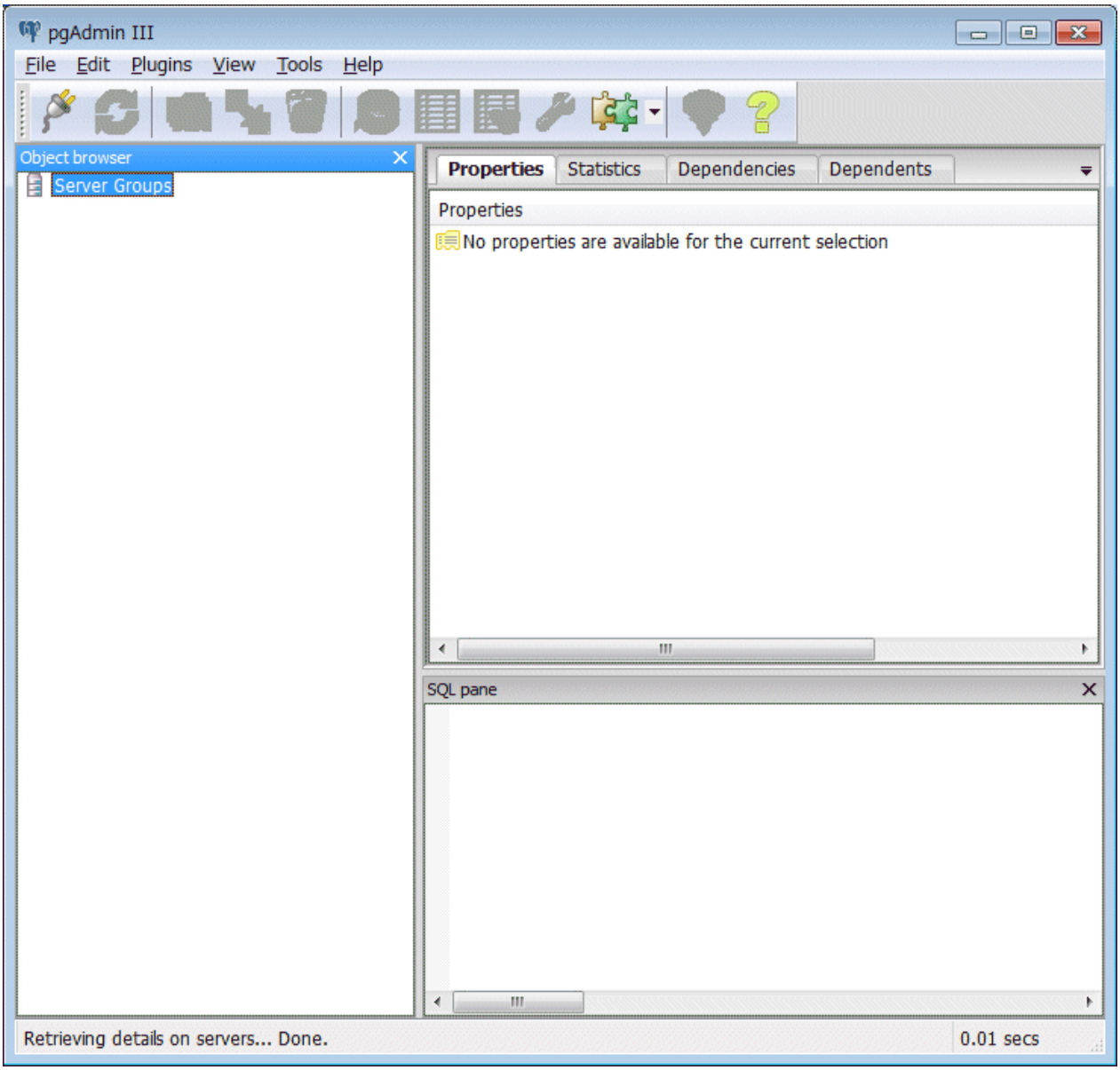
Windows(R) 8.1 or Windows Server(R) 2012 R2

From the [Apps] view, start [pgAdmin III(*AA*bit)(*x.y.z*)].

Other operating systems

Click [Start] >> [All Programs] >> [Enterprise Postgres Client (*AA*bit) *x.y.z*] and start [pgAdmin III(*AA*bit)(*x.y.z*)].

The following window is displayed when pgAdmin starts.



Note

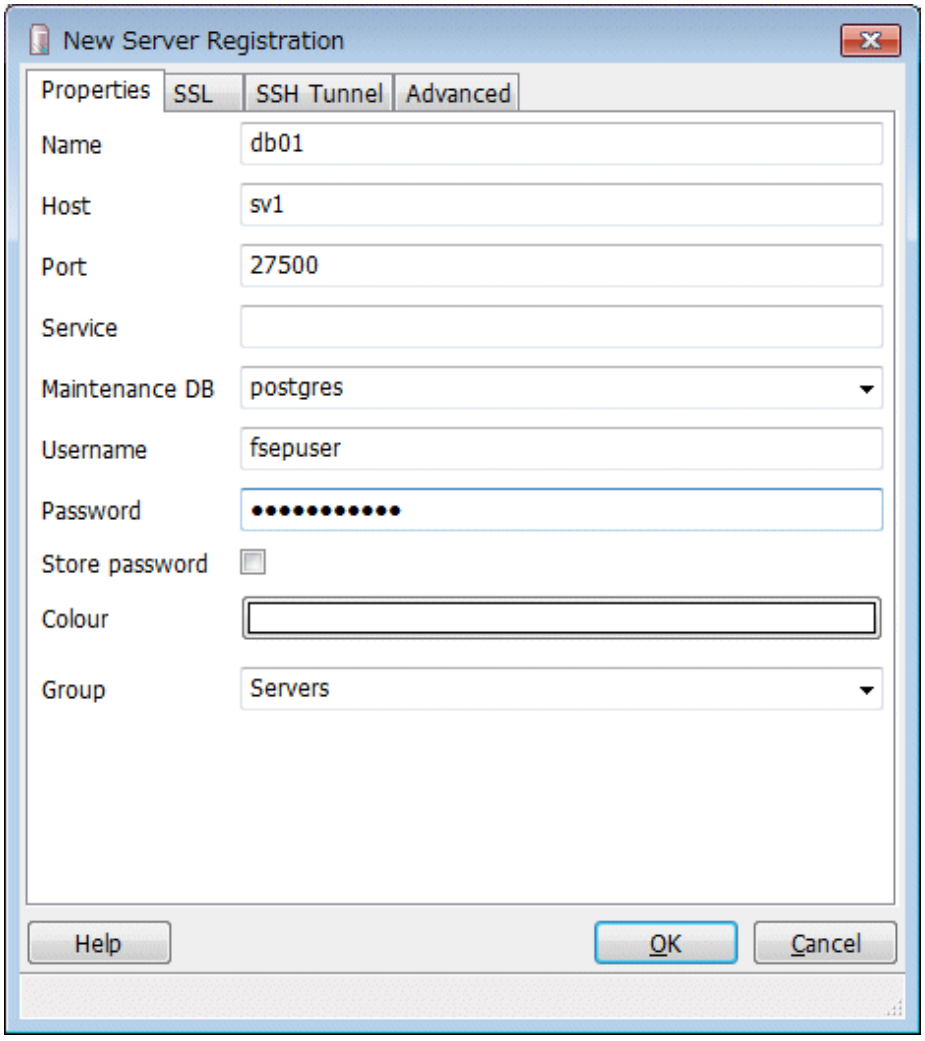
- You must start the instance to be connected to before using pgAdmin.
- Refer to "2.1 Starting and Stopping an Instance" for information on how to start an instance.
- Adobe(R) Reader(R) X is required for browsing the manual from [Enterprise Postgres Help] in pgAdmin.

1.3.2 Adding an Instance

This section describes how to add an instance to be connected to.

1. From the [File] menu in pgAdmin, click [Add Server].

2. In the [New Server Registration] window, specify a value for each item.



([Properties] tab)

- [Name]: Name of the instance to be managed
- [Host]: Host name or IP address of the server where Enterprise Postgres is installed
- [Port]: Port number of the instance
- [Username]: User ID of the instance administrator
- [Password]: Password for the user ID specified in [Username]

When you add an instance using pgAdmin, the instance is automatically connected to immediately after the addition is completed.

Note

If you select [Store password], a file storing the Enterprise Postgres connection password is created in the following location. Set the appropriate access permissions for the password file to protect it from unauthorized access.

- %APPDATA%\postgresql\pgpass.conf

1.3.3 Connecting/Disconnecting an Instance

This section describes how to connect pgAdmin to an instance, and how to disconnect it.

Note

To connect to an instance created using WebAdmin, you must first configure the settings in the [Client Authentication] window of WebAdmin to permit connection from pgAdmin.

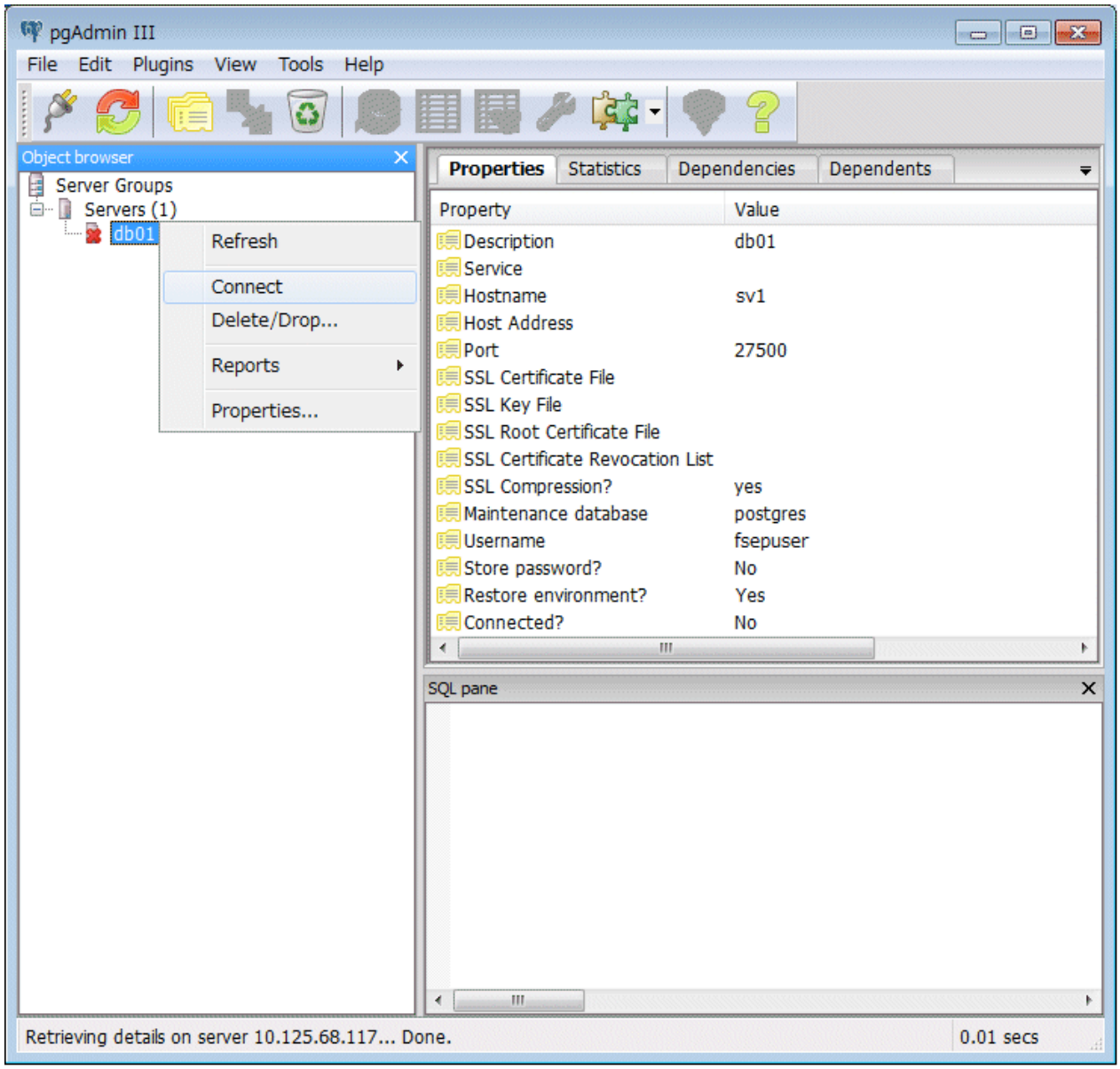
See

Refer to "Changing the settings" in the Installation and Setup Guide for Server for information on the [Client Authentication] window of WebAdmin.

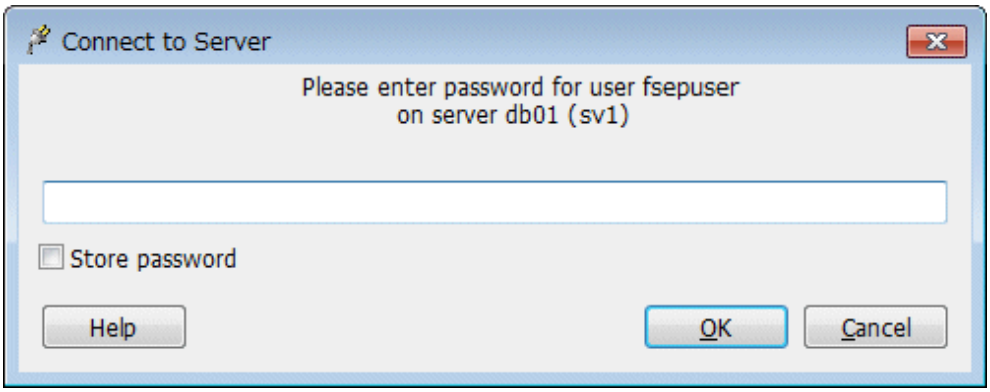
Connecting to an instance

Starting pgAdmin does not connect it to any instance.

To connect to an instance, right-click the instance in [Object browser] and select [Connect].



If a password was not saved when the instance was added, the following password entry window is displayed.



Disconnecting from an instance

To disconnect from an instance, right-click the server in [Object browser] in the pgAdmin window and select [Disconnect server].

1.4 Operations Using Commands

You can operate and manage the database using the following commands:

- Server commands

  This group of commands includes commands for creating a database cluster and controlling the database. You can run these commands on the server where the database is operating.

  To use these commands, you must configure the environment variables.

  See

  - Refer to "PostgreSQL Server Applications" under "Reference" in the PostgreSQL Documentation, or "Reference" for information on server commands.
  - Refer to "Configure the environment variables" under the procedure for creating an instance in "Using the initdb Command" in the Installation and Setup Guide for Server for information on the values to be set in the environment variables.

- Client commands

  This group of commands includes the psql command and commands for extracting the database cluster to a script file. These commands can be executed on the client that can connect to the database, or on the server on which the database is running.

  To use these commands, you need to configure the environment variables.

  See

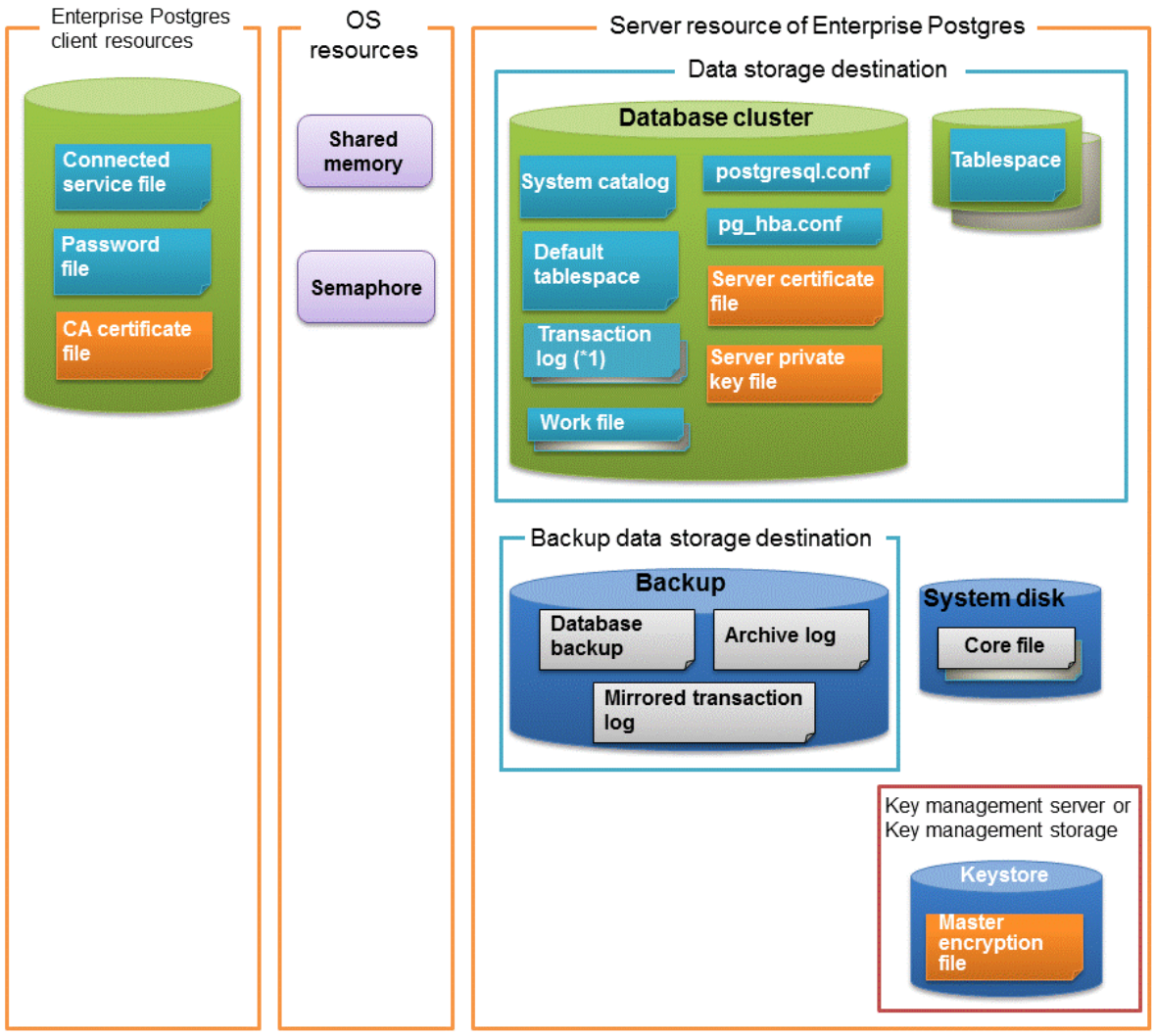
  - Refer to "PostgreSQL Client Applications" under "Reference" in the PostgreSQL Documentation, or "Reference" for information on client commands.
  - Refer to "Configuring Environment Variables" in the Installation and Setup Guide for Client for information on the values to be set in the environment variables.

1.5 Operating Environment of Enterprise Postgres

This section describes the operating environment and the file composition of Enterprise Postgres.

1.5.1 Operating Environment

The following figure shows the configuration of the Enterprise Postgres operating environment. The tables given below list the roles of the OS resources and Enterprise Postgres resources.



*1: To distribute the I/O load, place the transaction log on a different disk from the data storage destination.

Table 1.1 OS resources

Type	Role
Shared memory	Used when a database process exchanges information with an external process.
Semaphore	

Table 1.2 Enterprise Postgres client resources

Type	Role
Connection service file	Specifies information, such as the host name, user ID, and password, for connecting to Enterprise Postgres
Password file	Securely manages the password for connecting to Enterprise Postgres
CA certificate file	CA (certificate authority) certificate used for server authentication when encrypting communication data

Table 1.3 Server resources of Enterprise Postgres

Type	Role
Database cluster	Database storage area on the database storage disk. It is a collection of databases managed by an instance.
System catalog	Contains information required for the system to run, including the database definition information and the operation information created by the user
Default tablespace	Contains table files and index files stored by default
Transaction log	Contains log information in case of a crash recovery or rollback. This is the same as the WAL (Write Ahead Log).
Work file	Work file used when executing applications or commands
postgresql.conf	Contains information that defines the operating environment of Enterprise Postgres
pg_hba.conf	Enterprise Postgres uses this file to authenticate individual client hosts
Server certificate file	Contains information about the server certificate to be used when encrypting communication data and authenticating a server
Server private key file	Contains information about the server private key to be used when encrypting communication data and authenticating a server
Tablespace	Stores table files and index files in a separate area from the database cluster
Backup	Stores the data required for recovering the database when an error, such as disk failure, occurs
Database backup	Contains the backup data for the database
Archive log	Contains the log information for recovery.
Core file	Enterprise Postgres process core file that is output when an error occurs during an Enterprise Postgres process
Key management server or key management storage	Server or storage where the master encryption key file is located
Master encryption key file	Contains the master encryption key to be used when encrypting storage data. The master encryption key file is managed on the key management server or key management storage.

1.5.2 File Composition

Enterprise Postgres consists of the following files for controlling and storing the database. The table below shows the relationship between the number of such files and their location within a single instance.

Table 1.4 Number of files within a single instance and how to specify their location

File type	Required	Quantity	How to specify the location
Program files	Y	Multiple	**64-bit product** %Program Files%\Fujitsu\fsepserver64 **32-bit product (when installed on a 64-bit OS)** %Program Files(x86)%\Fujitsu\fsepserver32 **32-bit product (when installed on a 32-bit OS)** %Program Files%\Fujitsu\fsepserver32
Database cluster	Y	1	Specify using WebAdmin or server commands.
Tablespace	Y	Multiple	Specify using pgAdmin or the DDL statement.
Backup	Y	Multiple	Specify using WebAdmin or server commands.
Core file	Y	Multiple	Specify using WebAdmin, server commands, or postgresql.conf.

File type	Required	Quantity	How to specify the location
Server certificate file (*1)	N	1	Specify using postgresql.conf.
Server private key file (*1)	N	1	Specify using postgresql.conf.
Master encryption key file (*1)	N	1	Specify the directory created as the key store using postgresql.conf.
Connection service file (*1)	N	1	Specify using environment variables.
Password file (*1)	N	1	Specify using environment variables.
CA certificate file (*1)	N	1	Specify using environment variables.

Y: Mandatory
N: Optional

*1: Set manually when using the applicable feature.

Note

If anti-virus software is used, set scan exception settings for directories so that none of the files that comprise Enterprise Postgres are scanned for viruses. Alternatively, if the files that comprise Enterprise Postgres are to be scanned for viruses, stop Enterprise Postgres and perform the scan when tasks that use Enterprise Postgres are not operating.

1.6 Notes on Compatibility of Applications Used for Operations

When you upgrade Enterprise Postgres to a newer version, there may be some affect on applications due to improvements or enhancements in functionality.

Take this into account when creating applications so that you can maintain compatibility after upgrading to a newer version of Enterprise Postgres.

See

Refer to " Notes on Application Compatibility " in the Application Development Guide for details.

Chapter 2 Starting an Instance and Creating a Database

This chapter describes basic operations, from starting an instance to creating a database.

2.1 Starting and Stopping an Instance

This section describes how to start and stop an instance.

- 2.1.1 Using WebAdmin
- 2.1.2 Using Commands

Point

To automatically start or stop an instance when the operating system on the database server is started or stopped, refer to "Configuring Automatic Start and Stop of an Instance" in the Installation and Setup Guide for Server and configure the settings.

Note

The collected statistics are initialized if an instance is stopped in the "Immediate" mode or if it is abnormally terminated. To prepare for such initialization of statistics, consider regular collection of the statistics by using the SELECT statement. Refer to "The Statistics Collector" in "Server Administration" in the PostgreSQL Documentation for information on the statistics.

2.1.1 Using WebAdmin

WebAdmin enables you to start or stop an instance and check its operating status.

Starting an instance

Start an instance by using the [Monitor] window in WebAdmin.

The [Start] button is displayed when an instance is stopped.

To start a stopped instance, click [Start].

Stopping an instance

Stop an instance by using the [Monitor] window of WebAdmin.

The [Stop] button is displayed when an instance is active.

To stop an active instance, click [Stop].

Stop mode

Select the mode in which to stop the instance. The following describes the operations of the modes:

Stop mode	Connected clients	Backup being executed using the command
Smart mode (*1)	Waits for all connected clients to be disconnected.	Waits for backups being executed using the command to finish.
Fast mode	Rolls back all transactions being executed and forcibly disconnects clients.	Terminates backups being executed using the command.
Immediate mode	All server processes are terminated immediately. Crash recovery is executed the next time the instance is started.	

*1: When the processing to stop the instance in the Smart mode has started and you want to stop immediately, use the following procedure:

1. Restart the Web server feature of WebAdmin.
2. Log in to WebAdmin again.
3. Click the [Stop] button in the [Monitor] window, and select the Immediate mode to stop the instance.

Checking the operating status of an instance

You can check the operating status of an instance by using the [Monitor] window.

When an instance is started, "Started" is displayed as the operating status. When an instance is stopped, "Stopped" is displayed as the operating status. If an error is detected, an error message is displayed in the message list.

If an instance stops, remove the cause of stoppage and start the instance by using WebAdmin.

Figure 2.1 Status when an instance is active

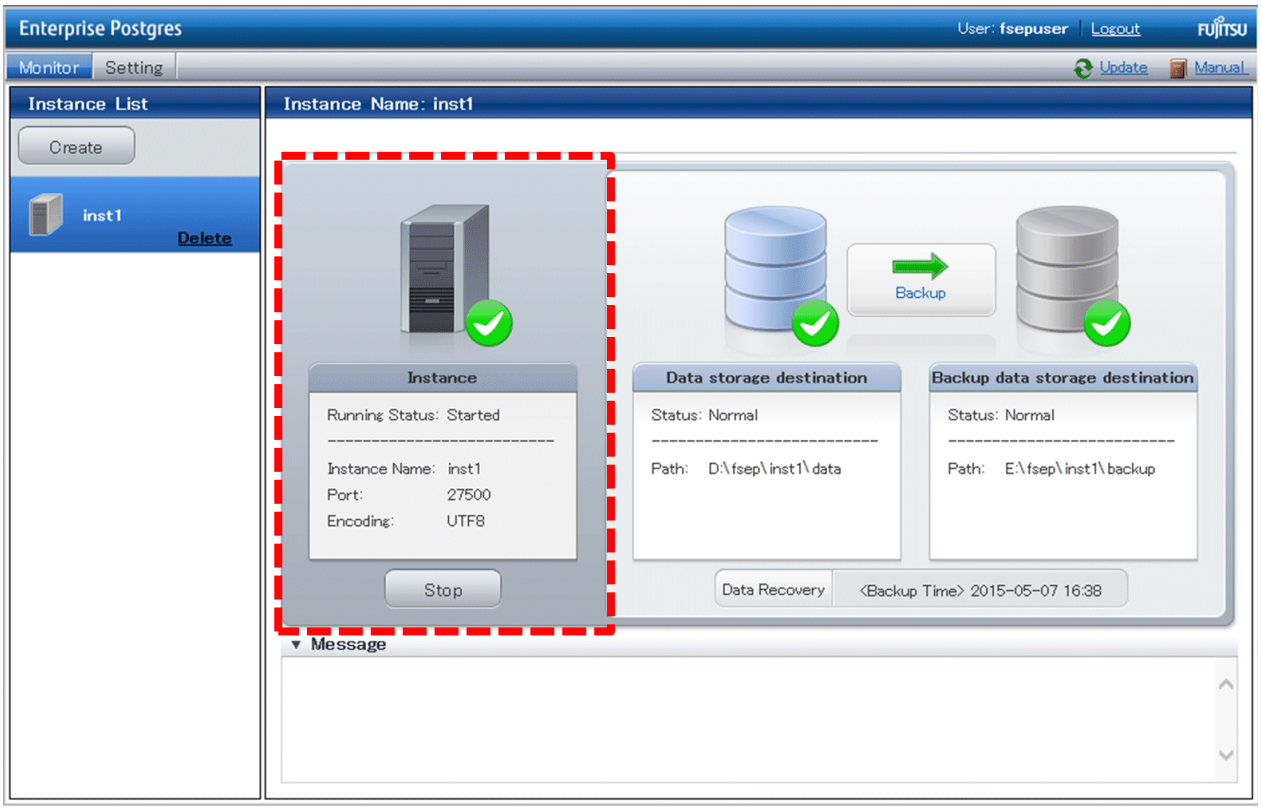
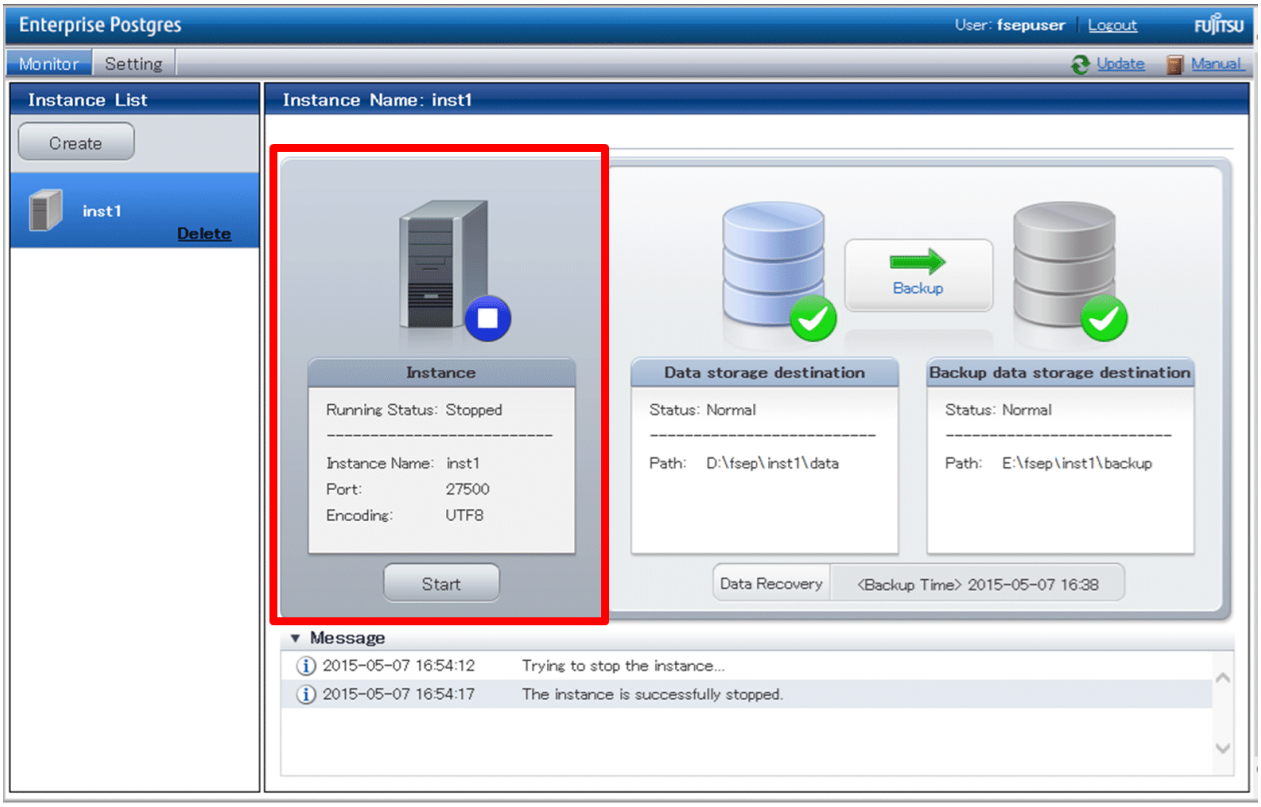


Figure 2.2 Status when an instance is stopped



Note

- If an error occurs while communicating with the server, there may be no response from WebAdmin. When this happens, close the browser and then log in again. If this does not resolve the issue, check the event log of the server and confirm whether a communication error has occurred.
- The following message is output during startup of an instance when the startup process is operating normally, therefore, the user does not need to be aware of this message:

```
FATAL:  the database system is starting up
```

2.1.2 Using Commands

The Windows service-related commands enable you to start or stop an instance and to check its operating state.

If you are to use Windows services, you should register instances in Windows services.

See

Refer to "When an instance was created with WebAdmin" in "Configuring Automatic Start and Stop of an Instance" in the Installation Guide for Server for information on registering instances in Windows services.

Note

While it is also possible for you to execute the pg_ctl command to start and stop instances without having to register instances in Windows services, it is recommended that you use Windows services to start and stop instances for the following reason:

- If you use the pg_ctl command to start an instance, the instance will be started as a user process. Therefore, when you close the [Command Prompt] window in which you executed the command, Windows forces the postgres process to stop.

Starting an instance

You can start an instance by specifying the service name in the net start command or sc start command.

Also, you can use the following procedure to start an instance in the Windows services window:

1. Display the [Services] window
   - Windows Server(R) 2012 and Windows Server(R) 2012 R2:

     In the [Start] screen, select [Administrative Tools], and then click [Services].
   - All other operating systems:

     In the [Start] menu, select [Administrative Tools], and then click [Services].
2. Start a service

   Select the instance name that you wish to start from the services list, and click [Start Service].

Stopping an instance

You can stop an instance by specifying the service name in the net stop command or sc stop command.

Also, you can use the following procedure to stop an instance in the Windows services window:

1. Display the [Services] window
   - Windows Server(R) 2012 and Windows Server(R) 2012 R2:

     In the [Start] screen, select [Administrative Tools], and then click [Services].
   - All other operating systems:

     In the [Start] menu, select [Administrative Tools], and then click [Services].
2. Stop the service

   Select the instance name that you wish to stop from the services list, and click [Stop Service]. If you stop a service while applications and commands are running, Enterprise Postgres will force those applications and commands to close and will stop normally.

Checking the operating state of an instance

Use the following procedure to check if an instance is operating correctly immediately after performing the operation to start an instance:

1. Display the [Services] window

   In the [Start] menu, select [Administrative Tools], and then click [Services].
2. Check the state of the service

   In the services list, check the state of the services for the applicable Enterprise Postgres.

To check the operating state of an instance during operation, use the pg_ctl command.

Specify the following in the pg_ctl command:

- Specify "status" as the mode.
- Specify the data storage destination directory in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

 Example

When the instance is active:

```
> pg_ctl status -D D:\database\inst1
pg_ctl: server is running (PID: 1234)
```

When the instance is inactive:

```
> pg_ctl status -D D:\database\inst1
pg_ctl: no server running
```

Information

You can also use the net start command or sc query command to check the operating state of an instance.

See

Refer to "pg_ctl" in "Reference" in the PostgreSQL Documentation for information on the pg_ctl command.

2.2 Creating a Database

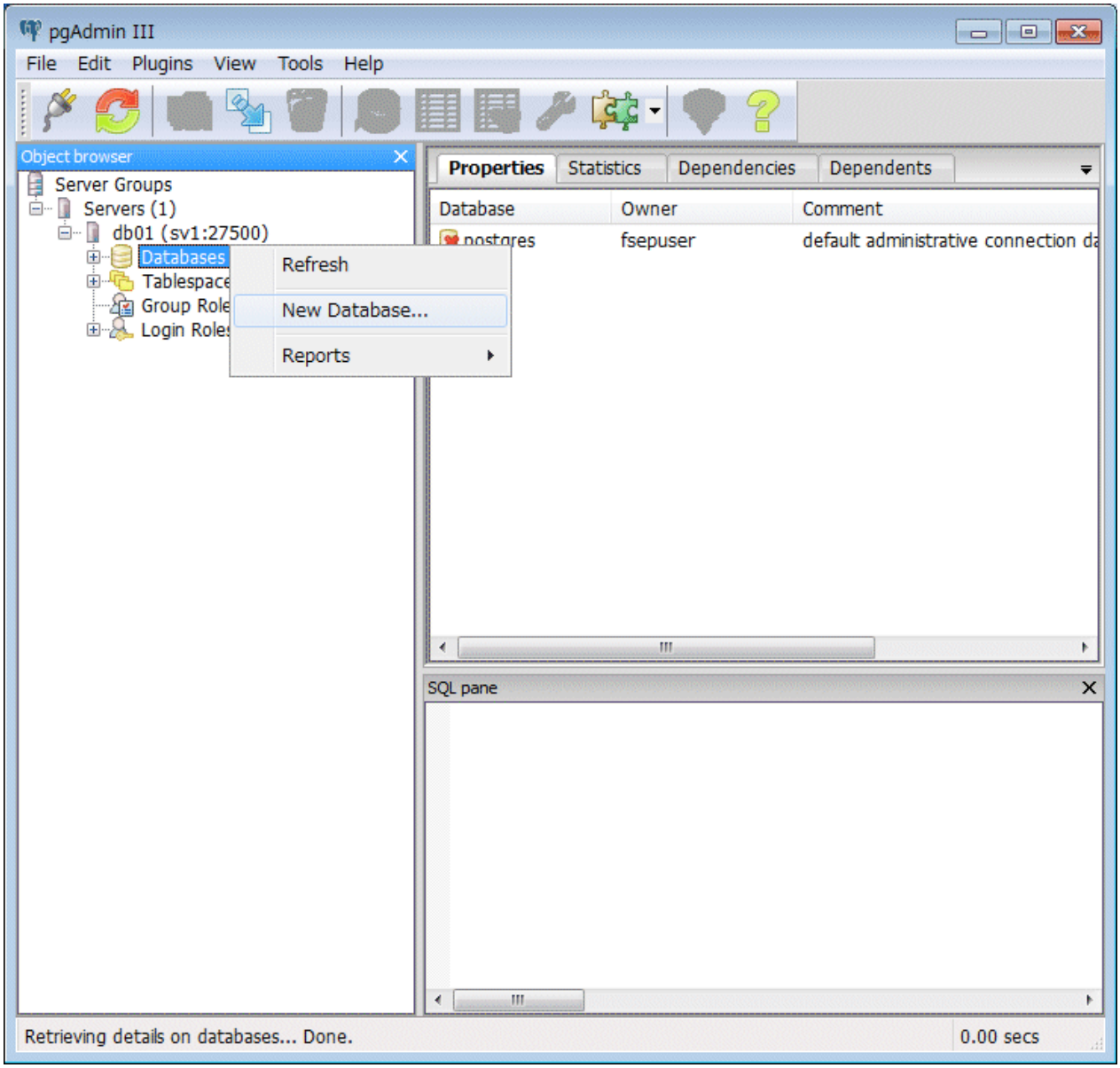
This section explains how to create a database.

- 2.2.1 Using pgAdmin
- 2.2.2 Using Client Commands

2.2.1 Using pgAdmin

Follow the procedure below to define a database using pgAdmin.

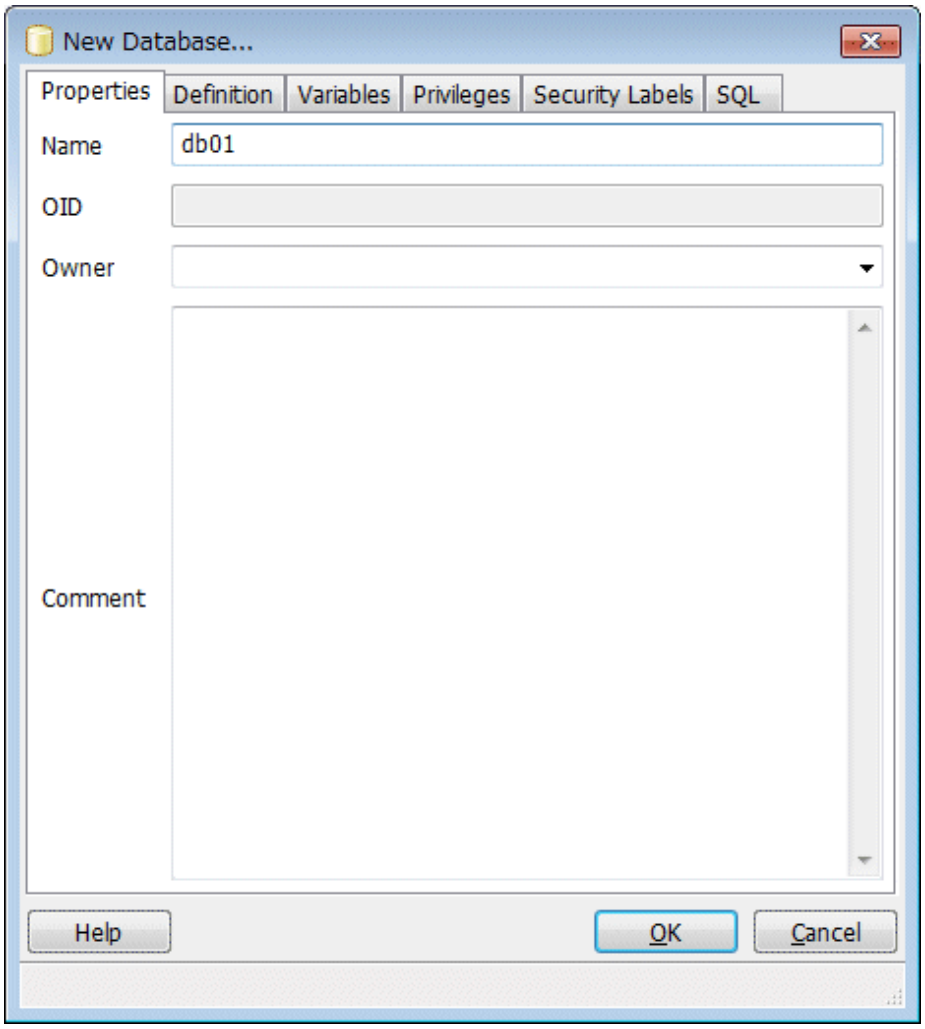
1. In the pgAdmin window, right-click [Database] in [Object browser], and then click [New Database] to display a new database window.



2. Specify appropriate values for the following items in the new database window.

   - [Properties] tab

     The following example illustrates creation of the database "db01".



   - [Name]: Name of the database to be managed

3. Click [OK] to create the database.

2.2.2 Using Client Commands

Follow the procedure below to define a database using client commands.

An example of operations on the server is shown below.

1. Use psql command to connect to the postgres database.
   Execute psql postgres.

```
> psql postgres
psql (9.4.4)
Type "help" for help.
```

2. Create the database.
   To create the database, execute the CREATE DATABASE databaseName; statement.

```
postgres=# CREATE DATABASE db01;
CREATE DATABASE
```

3. Confirm that the database is created.
   Execute the \l+ command, and confirm that the name of the database created in step 2 is displayed.

```
postgres=# \l+
```

4. Disconnect from the postgres database.
   Execute \q to terminate the psql command.

```
postgres=# \q
```

You can create a database using the createdb command.

See

Refer to "Creating a Database" in the "Tutorial" in the PostgreSQL Documentation for information on creating a database using the createdb command.

Chapter 3 Backing Up the Database

This chapter describes how to back up the database.

Backup methods

The following backup methods enable you to recover data to a backup point or to the state immediately preceding disk physical breakdown or data logical failure.

- Backup using WebAdmin

  This method enables you to back up data through intuitive window operations using the GUI.

  WebAdmin is used for recovery.

- Backup using the pgx_dmpall command

  Execute the pgx_dmpall command with a script to perform automatic backup.

  To back up data automatically, you must register the process in the automation software of the operating system. Follow the procedure given in the documentation for your operating system.

  The pgx_rcvall command is used for recovery.

Approximate backup time

The formula for deriving the approximate backup time when you use WebAdmin or the pgx_dmpall command is as follows:

```
backupTime = dataStorageDestinationUsage / diskWritePerformance x 1.5
```

- *dataStorageDestinationUsage*: Disk usage at the data storage destination
- *diskWritePerformance*: Maximum data volume (bytes/second) that can be written per second in the system environment where operation is performed
- 1.5: Coefficient to factor in tasks other than disk write (which is the most time-consuming step)

Note

- Use the selected backup method continuously.

  There are several differences, such as the data format, across the backup methods. For this reason, the following restrictions apply:

  - It is not possible to use one method for backup and another for recovery.
  - It is not possible to convert one type of backup data to a different type of backup data.

- There are several considerations for the backup of the keystore and backup of the database in case the data stored in the database is encrypted. Refer to the following for details:

  - 5.6.4 Backing Up and Recovering the Keystore
  - 5.7 Backing Up and Restoring/Recovering the Database

- If you have defined a tablespace, back it up. If you do not back it up, directories for the tablespace are not created during recovery, which may cause the recovery to fail. If the recovery fails, refer to the event log, create the tablespace, and then perform the recovery process again.

- If performing backups with WebAdmin, the following password file is temporarily created during backup for WebAdmin to connect to the database:

  - *userProfileFolder**localSettingsFolder*\Fujitsu\fsep_*version**instanceName*\pgpass.conf

  Therefore, when you are backing up corefiles created in the core_directory parameter of postgresql.conf, or log files created in the log_directory parameter of postgresql.conf, ensure not to back up the password files located in the same directories at the same time.

Information

The following methods can also be used to perform backup. Performing a backup using these methods allows you to restore to the point when the backup was performed.

- Backup using an SQL-based dump

  Dump the data by using SQL. This backup method also enables data migration.

- File system level backup

  This backup method requires you to stop the instance and use OS commands to backup database resources as files.

- Backup by continuous archiving

  This is the standard backup method for PostgreSQL.

Refer to "Backup and Restore" in "Server Administration" in the PostgreSQL Documentation for information on these backup methods.

3.1 Periodic Backup

It is recommended that you perform backup periodically.

Backing up data periodically using WebAdmin or the pgx_dmpall command has the following advantages:

- This method reduces disk usage, because obsolete archive logs (transaction logs copied to the backup data storage destination) are deleted. It also minimizes the recovery time when an error occurs.

Backup cycle

The time interval when backup is performed periodically is called the backup cycle. For example, if backup is performed every morning, the backup cycle is 1 day.
The backup cycle depends on the jobs being run, but on Enterprise Postgres it is recommended that operations are run with a backup cycle of at least once per day.

3.2 Backup Methods

This section describes the methods for backing up the database.

3.2.1 Using WebAdmin

You can use WebAdmin to perform backup and check the backup status.

Note

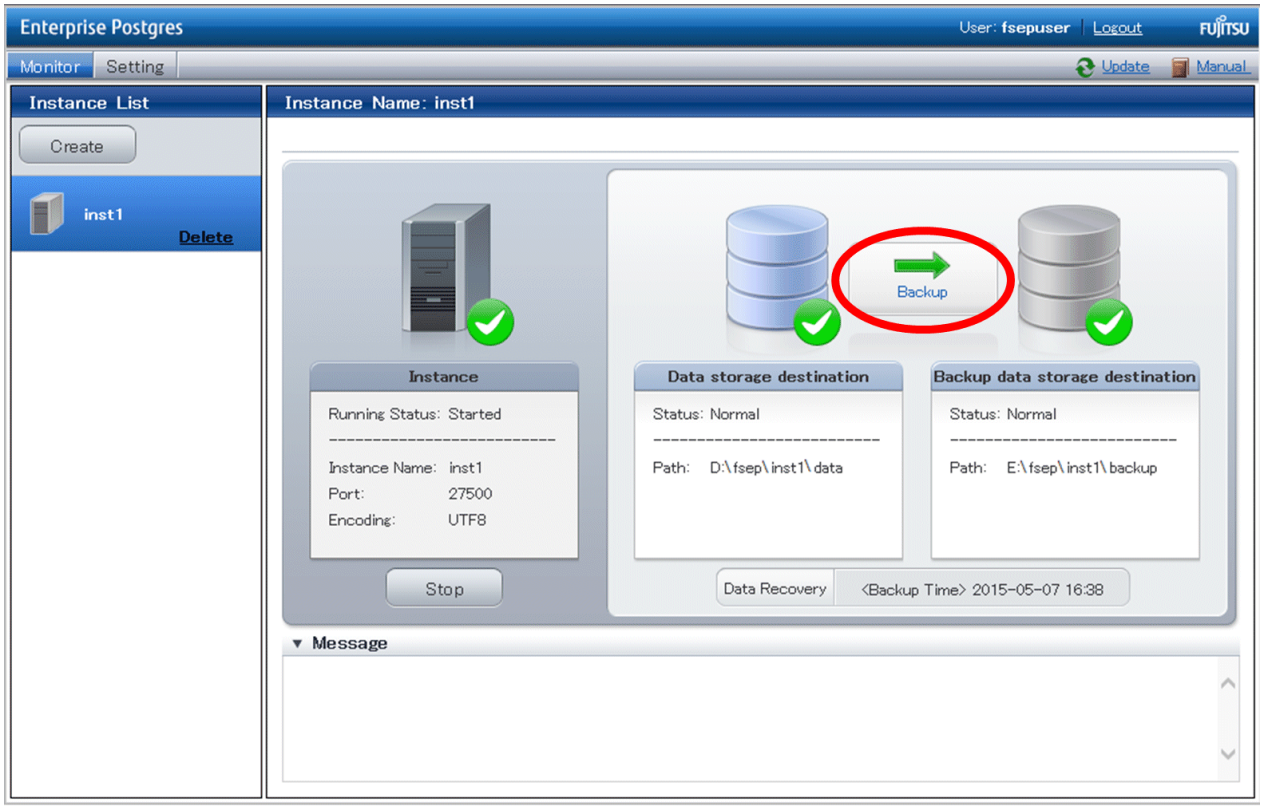
If the data to be stored in the database is to be encrypted, it is necessary to enable the automatic opening of the keystore before doing so. Refer to "5.6.3 Enabling Automatic Opening of the Keystore" for details.

Backup operation

Follow the procedure below to back up the database.

1. Select database backup

   In the [Monitor] window of WebAdmin, click [->] marked "Backup".



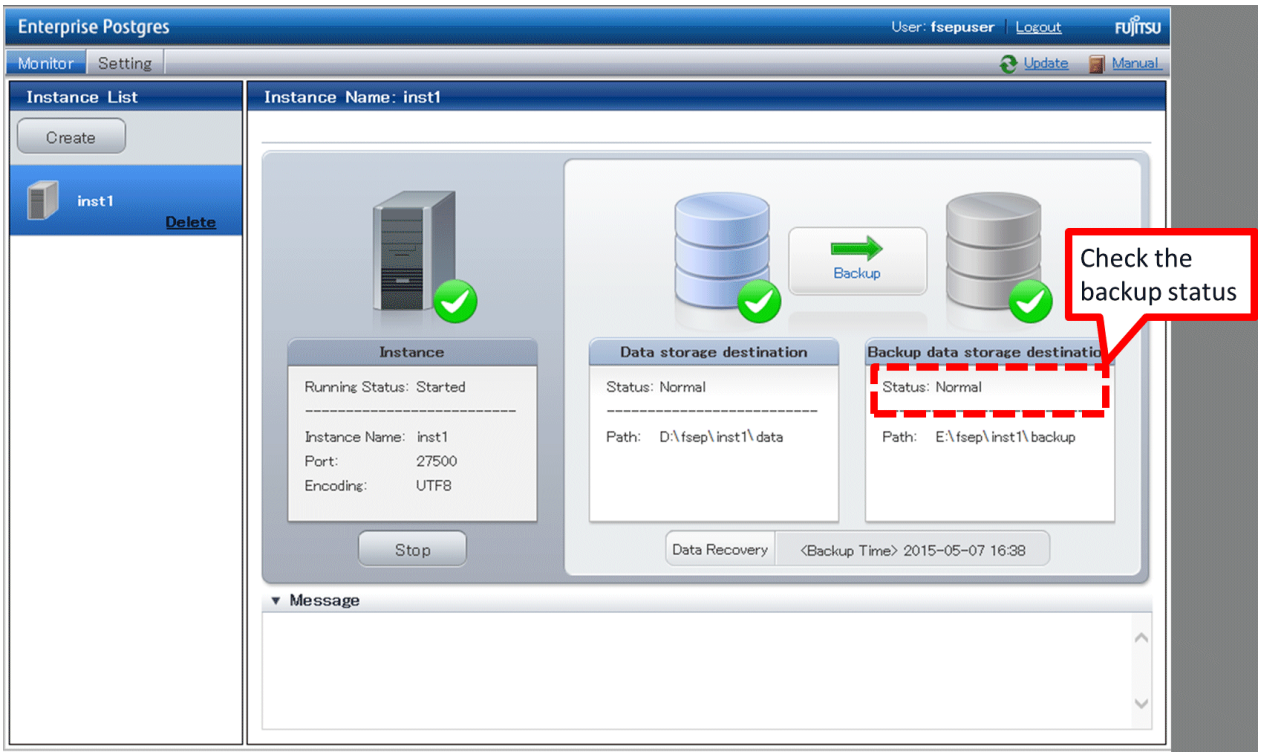
2. Back up the database

   The [Backup] dialog box is displayed. To perform backup, click [Run].
   An instance is automatically started when backup is performed.

Backup status

If an error occurs and backup fails, [Error] is displayed adjacent to [Status] under [Data storage destination] or [Backup data storage destination] in the [Monitor] window. An error message is also displayed in the message list.

In this case, the backup data is not optimized. Ensure that you check the backup result whenever you perform backup. If backup fails, [Solution] appears to the right of the error message. Clicking this button displays information explaining how to resolve the cause of the error. Remove the cause of failure, and perform backup again.



Note

If the data to be stored in the database is to be encrypted, it is necessary to enable the automatic opening of the keystore before doing so. Refer to "5.6.3 Enabling Automatic Opening of the Keystore" for details.

3.2.2 Using Server Commands

Use the pgx_dmpall command and pgx_rcvall command to perform backup and check the backup result.

Preparing for backup

You must prepare for backup before actually starting the backup process.

Follow the procedure below.

See

Refer to " Preparing Directories to Deploy Resources" in the Installation and Setup Guide for Server for information on the location of directories required for backup and for points to take into account.

1. Prepare the backup data storage disk

   For backup, prepare a separate disk unit from the database storage disk and mount it using the operating system commands.

2. Create a directory where the backup data will be stored

   Create an empty directory.

   In [Properties] in Windows(R) Explorer, set appropriate permissions so that only the instance administrator can access the directory.

   ## See

   Refer to [Help and Support] in Windows(R) for information on [Properties].

3. Specify the settings required for backup

   Stop the instance, and set the following parameters in the postgresql.conf file.

   Start the instance after editing the postgresql.conf file.

Parameter name	Setting	Description
backup_destination	Name of the directory where the backup data will be stored	Specify the name of the directory where the backup data will be stored. Appropriate privileges that allow only the instance administrator to access the directory must already be set. Place the backup data storage destination directory outside the data storage destination directory, the tablespace directory, and the transaction log storage destination directory.
wal_level	archive or hot_standby(*1)	Specify the output level for the transaction log. *1: hot_standby is a setting for streaming replication.
archive_mode	on	Specify the archive log mode. Specify [on] (execute).
archive_command	'cmd /c ""*installationDirectory*\\bin\\pgx_xlogcopy.cmd" "%p" "*backupDataStorageDestinationDirectory*\\archived_xlog\\%f"""	Specify the path name of the command that will save the transaction log and the storage destination. Note the following when specifying the path: - Specify \\ as the path delimiter. - Enclose the path in double quotes ("") if it contains spaces.

   Refer to "Appendix A Parameters" and "Write Ahead Log" under "Server Administration" in the PostgreSQL Documentation for information on the parameters.

Backup operation

Use the pgx_dmpall command to perform backup. You can even embed the pgx_dmpall command in OS automation software to perform backup.

The backup data is stored in the directory specified in the backup_destination parameter of postgresql.conf.

Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

Example

```
> pgx_dmpall -D D:\database\inst1
```

Note

Backup stores the data obtained during the backup and the backup data of the data obtained during previous backup.

If the data to be stored in the database is encrypted, refer to the following and back up the keystore:

- 5.6.4 Backing Up and Recovering the Keystore

Backup status

Use the pgx_rcvall command to check the backup status.

Specify the following values in the pgx_rcvall command:

- The -l option indicates backup data information.
- Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

```
> pgx_rcvall -l -D D:\database\inst1
Date                    Status          Dir
2015-05-01 13:30:40     COMPLETE        E:\backup\inst1\2015-05-01_13-30-40
```

If an error occurs and backup fails, a message is output to the event log.

In this case, the backup data is not optimized. Ensure that you check the backup result whenever you perform backup. If backup fails, remove the cause of failure and perform backup again.

See

Refer to "pgx_dmpall" and "pgx_rcvall" in the Reference for information on the pgx_dmpall command and pgx_rcvall command.

Setting a restore point

In case you want to recover your database to a certain point in time, you can name this particular point in time, which is referred to as the restore point, by using the psql command.

By setting a restore point before executing an application, it becomes easy to identify up to which point in time the data will be reverted.

A restore point can be set to any point in time after a backup is executed. However, if a restore point is set before a backup is executed, the database cannot be recovered to that point in time. This is because restore points are recorded in the archive logs, and the archive logs are discarded when backups are executed.

Example

The following example uses the psql command to connect to the database and execute the SQL statement to set a restore point.

However, when considering continued compatibility of applications, do not use functions directly in SQL statements. Refer to "Notes on Application Compatibility" in the Application Development Guide for details.

```
postgres=# SELECT pg_create_restore_point('batch_20150503_1');
LOG:  restore point "batch_20150503_1" created at 0/20000E8
STATEMENT:  select pg_create_restore_point('batch_20150503_1');
 pg_create_restore_point
-------------------------
 0/20000E8
(1 row)
```

Refer to "8.3.2 Using the pgx_rcvall Command" for information on using a restore point to recover the database.

Note

- Name restore points so that they are unique within the database. Add the date and time of setting a restore point to distinguish it from other restore points, as shown below:
  - YYMMDD_HHMMSS
    - YYMMDD: Indicates the date
    - HHMMSS: Indicates the time
- There is no way to check restore points you have set. Keep a record in, for example, a file.

See

Refer to "System Administration Functions" under "Functions and Operators" in the PostgreSQL Documentation for information on pg_create_restore_point.

Chapter 4 Configuring Secure Communication Using Secure Sockets Layer

If communication data transferred between a client and a server contains confidential information, encrypting the communication data can protect it against threats, such as eavesdropping on the network.

4.1 Configuring Communication Data Encryption

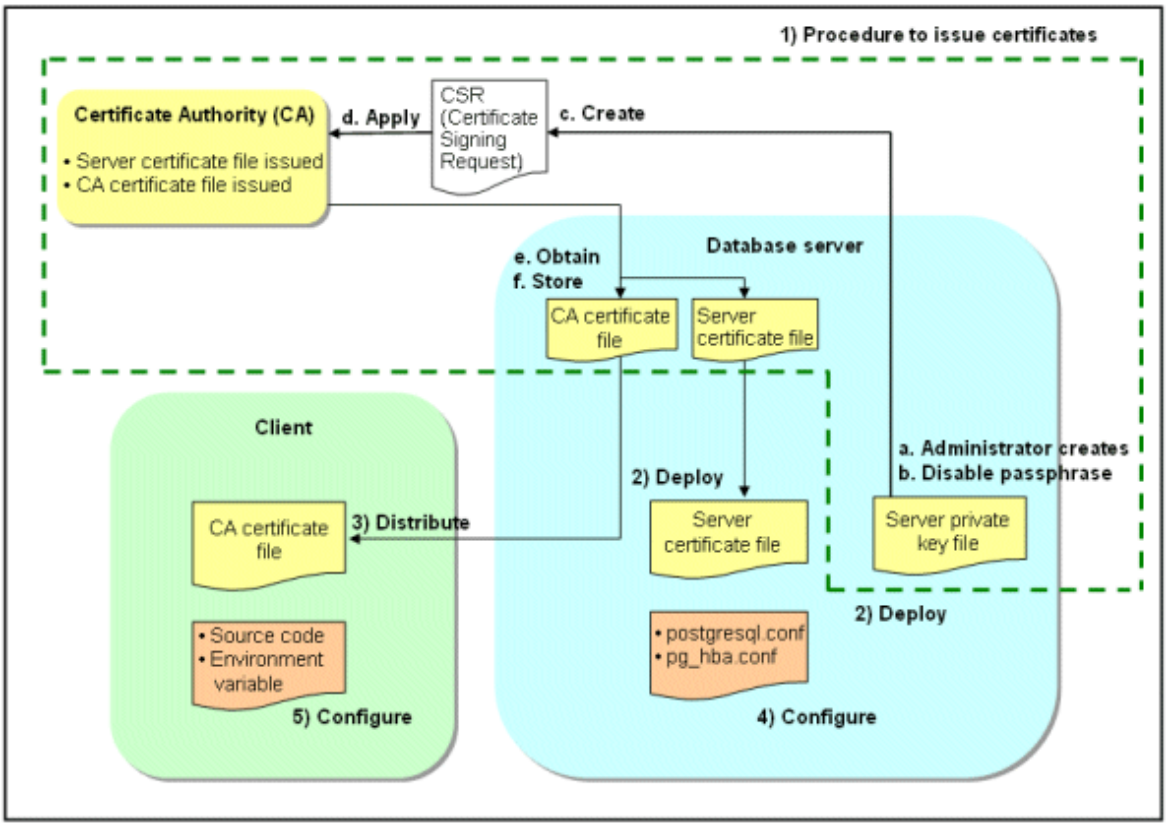
To encrypt communication data transferred between a client and a server, configure communication data encryption as described below. Communication data encryption not only protects the communication content, but it also guards against man-in-the-middle (MITM) attacks (for example, data and password theft through server impersonation).

Table 4.1 Configuration procedure

Configuration procedure
1) Issue a certificate
2) Deploy a server certificate file and a server private key file
3) Distribute a CA certificate file to the client
4) Configure the operating environment for the database server
5) Configure the operating environment for the client

The following figure illustrates the environment for communication data encryption.

Figure 4.1 Environment for communication data encryption



4.1.1 Issuing a Certificate

For authenticating servers, you must acquire a certificate issued by the certificate authority (CA).
Enterprise Postgres supports X.509 standard PEM format files. If the certificate authority issues a file in DER format, use a tool such as the openssl command to convert the DER format file to PEM format.

The following provides an overview of the procedure. Refer to the procedure published by the public or independent certificate authority (CA) that provides the certificate file for details.

a. Create a server private key file

b. Disable the passphrase for the server private key file

c. Create a CSR (signing request for obtaining a server certificate) from the server private key file

d. Apply to the certificate authority (CA) for a server certificate

e. Obtain a server certificate file and a CA certificate file from the certificate authority (CA)

f. Store the server certificate file and the CA certificate file
   Note: If you lose or destroy the certificates, you will need to have them re-issued.

The above procedure enables you to prepare the following files:

- Server private key file
- Server certificate file
- CA certificate file

4.1.2 Deploying a Server Certificate File and a Server Private Key File

Create a directory on the local disk of the database server and store the server certificate file and the server private key file in it.
Use the operating system features to set access privileges for the server certificate file and the server private key file so that only the database administrator has load privileges.
Back up the server certificate file and the server private key file in the event that data corruption occurs and store them securely.

4.1.3 Distributing a CA Certificate File to the Client

Create a directory on the local disk of the client and place the distributed CA certificate file there. Use the operating system features to set load privileges to protect the CA certificate file against accidental deletion.

4.1.4 Configuring the Operating Environment for the Database Server

See

Refer to "Secure TCP/IP Connections with SSL" under "Server Administration" in the PostgreSQL Documentation for details.

4.1.5 Configuring the Operating Environment for the Client

See

Refer to the following sections in the Application Development Guide for details, depending on your application development environment:

- "Settings for Encrypting Communication Data" under "Setup" in "JDBC Driver"
- "Settings for Encrypting Communication Data" under "Setup" in "C Library (libpq)"
- "Settings for Encrypting Communication Data" under "Setup" in "Embedded SQL in C"

4.1.6 Performing Database Multiplexing

When you perform communication that uses database multiplexing and a Secure Socket Layer server certificate, certificates with the same "Common Name" must be used. To ensure this, take one of the following actions:

- Create one server certificate, replicate it, and place a copy on each server used for database multiplexing.
- Create a server certificate with the same "Common Name" for each server used for database multiplexing.

 See

Refer to "Using the Application Connection Switch Feature" in the Application Development Guide for information on how to specify applications on the client.

Chapter 5 Protecting Storage Data Using Transparent Data Encryption

This chapter describes how to encrypt data to be stored in the database.

5.1 Protecting Data Using Encryption

With PostgreSQL, data in a database is protected from access by unauthorized database users through the use of authentication and access controls. However, the OS file is not protected from attackers who bypass the database server's authentication and access controls.

With Enterprise Postgres, data inside the OS file is encrypted, so valuable information is protected even if the file or disk is stolen.

Data to be stored in a database is encrypted when it is written to the data file, and decrypted when it is read.

This is performed automatically by the instance, so the user and the application need not be aware of key management and encryption or decryption. This process is called TDE (Transparent Data Encryption).

The characteristics of TDE are described below.

Encryption mechanisms

Two-layer encryption key and the keystore

In each tablespace, there is a tablespace encryption key that encrypts and decrypts all the data within. The tablespace encryption key is encrypted by the master encryption key and saved.
Only one master encryption key exists in a database cluster. It is encrypted based on a passphrase specified by the user and stored in a keystore. Enterprise Postgres provides a file-based keystore. Attackers who do not know the passphrase cannot read the master encryption key from the keystore.

Strong encryption algorithms

TDE uses the Advanced Encryption Standard (AES) as its encryption algorithm. AES was adopted as a standard in 2002 by the United States Federal Government, and is used throughout the world.

Faster encryption and decryption based on hardware

TDE minimizes the overhead of encryption and decryption by using the AES-NI (Advanced Encryption Standard New Instructions) built into Intel(R) Xeon(R) processors since the 5600 series. This means that even in situations where previously the minimum encryption target was selected as a tradeoff between performance and security, it is now possible to encrypt all the data of an application.

You can reference a list of processors equipped with AES-NI on the following page at Intel Corporation's website:

http://ark.intel.com/search/advanced/?s=t&AESTech=true

Zero overhead storage areas

Encryption does not change the size of data stored in tables, indexes, or WAL. There is, therefore, no need for additional estimates or disks.

Scope of encryption

All user data within the specified tablespace

The tablespace is the unit for specifying encryption. All tables, indexes, temporary tables, and temporary indexes created in the encrypted tablespace are encrypted. There is no need for the user to consider which tables and strings to encrypt.

Backup data

The pgx_dmpall command and pg_basebackup command create backup data by copying the OS file. Backups of the encrypted data are, therefore, also encrypted. Information is protected from leakage even if the backup medium is stolen.

WAL and temporary files

WAL, which is created by updating encrypted tables and indexes, is encrypted with the same security strength as the update target. When large merges and sorts are performed, the encrypted data is written to a temporary file in encrypted format.

Streaming replication support

You can combine streaming replication and transparent data encryption. The data and WAL encrypted on the primary server is transferred to the standby server in its encrypted format and stored.

Note

The following are not encrypted:

- pg_dump and pg_dumpall output files
- Files output by the COPY command
- Notification event payloads that communicate using the LISTEN or NOTIFY command

5.2 Setting the Master Encryption Key

To use transparent data encryption, you must create a keystore and set the master encryption key.

1. In the keystore_location parameter of postgresql.conf, specify the directory to store the keystore.

   Specify a different location for each database cluster.

   ```
   keystore_location = 'C:\\key\\store\\location'
   ```

   Refer to "Appendix A Parameters" for information on postgresql.conf.

   After editing the postgresql.conf file, either start or restart the instance.

   - Using WebAdmin

     Refer to "2.1.1 Using WebAdmin", and restart the instance.

   - Using commands

     Refer to "2.1.2 Using Commands", and restart the instance.

2. Execute an SQL function, such as the one below, to set the master encryption key. This must be performed by the superuser. Execute it as the database superuser.

   ```
   SELECT pgx_set_master_key('passphrase');
   ```

   The value "passphrase" is the passphrase that will be used to open the keystore. The master encryption key is protected by this passphrase, so avoid specifying a short simple string that is easy to guess.

   Refer to "B.2 Transparent Data Encryption Control Functions" for information on the pgx_set_master_key function.

Note

Note that if you forget the passphrase, you will not be able to access the encrypted data. There is no method to retrieve a forgotten passphrase and decrypt data. Do not, under any circumstances, forget the passphrase.

The pgx_set_master_key function creates a file with the name keystore.ks in the keystore storage destination. It also creates a master encryption key from random bit strings, encrypts it with the specified passphrase, and stores it in keystore.ks. At this point, the keystore is open.

5.3 Opening the Keystore

To create encrypted tablespaces and access the encrypted data, you must first open the keystore. When you open the keystore, the master encryption key is loaded into the database server memory and becomes usable for encryption and decryption.

You need to open the keystore each time you start the instance. To open the keystore, the database superuser must execute the following SQL function.

```
SELECT pgx_open_keystore('passphrase');
```

The value "passphrase" is the passphrase specified during creation of the keystore.

Refer to "B.2 Transparent Data Encryption Control Functions" for information on the pgx_open_keystore function.

Note that, in the following cases, the passphrase must be entered when starting the instance, because the encrypted WAL must be decrypted for recovery. In this case, the above-mentioned pgx_open_keystore function cannot be executed.

- If performing crash recovery at the time of starting the instance
- If performing recovery using continuous archiving

For the above cases, select one of the following methods:

- Use an automatically opening keystore

  Select this method if ease of operation has priority over enhanced security. When using an automatically opening keystore, the content of the keystore file is decrypted and a copy of the keystore file is generated. Although the content of this file is obfuscated, the level of security becomes slightly weaker.

  Select this method if performing operations using WebAdmin.

- Enter a passphrase when starting an instance

  Select this method if enhanced security has priority over ease of operation.

  Specify the --keystore-passphrase in the pg_ctl command and start the instance. This displays the prompt that asks for the passphrase to be entered.

  ```
  > pg_ctl --keystore-passphrase start
  Enter the passphrase:
  The server is currently initiating
  >
  ```

  After performing the above operation, use the pg_ctl command to stop the instance.

  Then start the instance in Windows services. Refer to "2.1.2 Using Commands" for information on how to start an instance in Windows services.

Point

When using an automatically opening keystore, you do not need to enter the passphrase and you can automatically open the keystore when the database server starts. Refer to "5.6.3 Enabling Automatic Opening of the Keystore" for details.

5.4 Encrypting a Tablespace

The keystore must be open before you can create an encrypted tablespace.

When creating a tablespace that will be encrypted, configure the encryption algorithm in the runtime parameters. For example, to create a tablespace with the name secure_tablespace using AES with a key length of 256 bits as the encryption algorithm, configure as shown below.

```
-- Specify the encryption algorithm for the tablespace to be created below
SET tablespace_encryption_algorithm = 'AES256';
CREATE TABLESPACE secure_tablespace LOCATION 'C:\My\Data\Dir';
-- Specify that the tablespace to be created below is not to be encrypted
SET tablespace_encryption_algorithm = 'none';
```

Or

```
CREATE TABLESPACE secure_tablespace LOCATION '\My\Data\Dir' tablespace_encryption_algorithm =
'AES256';
```

You can use AES with a key length of 128 bits or 256 bits as the encryption algorithm. It is recommended that you use 256-bit AES. Refer to "Appendix A Parameters" for information on how to specify the runtime parameters.

If user provides both GUC and command line options while creating the tablespace, the preference is given to the command line option.

The pg_default and pg_global tablespaces cannot be encrypted.

Create tables and indexes in the encrypted tablespace that you created. Relations created in the encrypted tablespace are automatically encrypted.

Example

Example 1: Specifying an encrypted tablespace when creating it

```
CREATE TABLE my_table (...)
    TABLESPACE secure_tablespace;
```

Example 2: Not explicitly specifying a tablespace when creating it and instead using the default tablespace

```
SET default_tablespace = 'secure_tablespace';
CREATE TABLE my_table (...);
```

The process is the same for encrypting temporary tables and temporary indexes. In other words, either explicitly specify the TABLESPACE clause or list encrypted tablespaces in the temp_tablespaces parameter, and then execute CREATE TEMPORARY TABLE or CREATE INDEX.

If you specify an encrypted tablespace in the TABLESPACE clause of the CREATE DATABASE statement when creating a database, relations that you create in the database without explicitly specifying a tablespace will be encrypted. Furthermore, the system catalog is also encrypted, so the source code of user-defined functions is also protected.

Note

An encrypted tablespace cannot be created from the window used for creating the pgAdmin tablespace, or from the query tool. To create an encrypted tablespace, click [PSQL Console] from the [Plugins] menu and create an encrypted tablespace in the psql console window.

5.5 Checking an Encrypted Tablespace

The pgx_tablespaces system view displays information about whether each tablespace has been encrypted, and about the encryption algorithm. Refer to "C.1 pgx_tablespaces" for information on strings.

You can discover which tablespaces have been encrypted by executing the following SQL statements.

However, when considering continued compatibility of applications, do not reference system catalogs (pg_tablespace) directly in SQL statements.

```
SELECT spcname, spcencalgo
FROM pg_tablespace ts, pgx_tablespaces tsx
WHERE ts.oid = tsx.spctablespace;
```

Example

```
postgres=# SELECT spcname, spcencalgo FROM pg_tablespace ts, pgx_tablespaces tsx WHERE ts.oid =
tsx.spctablespace;
      spcname      | spcencalgo
-------------------+------------
 pg_default        | none
 pg_global         | none
 secure_tablespace | AES256
(3 rows)
```

See

Refer to "Notes on Application Compatibility" in the Application Development Guide for information on how to maintain application compatibility.

5.6 Managing the Keystore

This section describes how to manage the keystore and the master encryption key to guard against the threat of theft.

5.6.1 Changing the Master Encryption Key

Using the same encryption key for an extended period gives attackers an opportunity to decipher the encrypted data. It is recommended that you change the key at regular intervals, or whenever the key is exposed to risk.

Adhere to the industry's best practices for encryption algorithms and key management when considering how often the key should be changed. For example, the NIST in the United States has published "NIST Special Publication 800-57". The PCI DSS also refers to this publication. This publication recommends changing the master encryption key once a year.

To change the master encryption key, execute the pgx_set_master_key function, which is the same function used for configuring the key. Refer to "5.2 Setting the Master Encryption Key" for details.

After changing the master encryption key, you must immediately back up the keystore.

5.6.2 Changing the Keystore Passphrase

In security policies for organizations, it is usually a requirement that the passphrase be changed whenever a security administrator who knows the passphrase is removed from duties due to transfer or retirement. It is also recommended that the passphrase be changed if it is ever exposed to risks due to deception such as social engineering.

To change the keystore passphrase, execute the following SQL function as a superuser.

```
SELECT pgx_set_keystore_passphrase('oldPassphrase', 'newPassphrase');
```

After changing the passphrase, you must immediately back up the keystore.

Refer to "B.2 Transparent Data Encryption Control Functions" for information on the pgx_set_keystore_passphrase function.

5.6.3 Enabling Automatic Opening of the Keystore

When using an automatically opening keystore, you do not need to enter the passphrase and you can automatically open the keystore when the instance starts. Execute the pgx_keystore command to enable automatic opening of the keystore.

```
> pgx_keystore --enable-auto-open C:\key\store\location\keystore.ks
Enter the passphrase:
Automatic opening of the keystore is now enabled
>
```

See

Refer to "pgx_keystore" in the Reference for information on pgx_keystore command.

When automatic opening is enabled, an automatically opening keystore is created in the same directory as the original keystore. The file name of the automatically opening keystore is keystore.aks. The file keystore.aks is an obfuscated copy of the decrypted content of the keystore.ks file. As long as this file exists, there is no need to enter the passphrase to open the keystore when starting the instance.

Do not delete the original keystore file, keystore.ks. It is required for changing the master encryption key and the passphrase. When you change the master encryption key and the passphrase, keystore.aks is recreated from the original keystore file, keystore.ks.

Protect keystore.ks, keystore.aks, and the directory that stores the keystore so that only the user who starts the instance can access them.

Configure the permission of the files so that only the user who starts the instance can access the SQL functions and commands that create these files. Accordingly, manually configure the same permission mode if the files are restored.

Set the permission mode in [Properties] in Windows(R) Explorer.

See

Refer to [Help and Support] in Windows(R) for information on [Properties].

An automatically opening keystore will only open on the computer where it was created.

To disable automatic opening of the keystore, delete keystore.aks.

Note

- To use WebAdmin for recovery, you must enable automatic opening of the keystore.
- Refer to "5.7 Backing Up and Restoring/Recovering the Database" after enabling or reconfiguring encryption to back up the database.
- Specify a different directory from those below as the keystore storage destination:
  - Data storage destination
  - Tablespace storage destination
  - Transaction log storage destination
  - Backup data storage destination

5.6.4 Backing Up and Recovering the Keystore

Back up the keystore at the following times in case it is corrupted or lost. Note that you must store the database and the keystore on separate data storage media. Storing both on the same data storage medium risks the danger of the encrypted data being deciphered if the medium is stolen. A passphrase is not required to open an automatically opening keystore, so store this type of keystore in a safe location.

- When the master encryption key is first configured
- When the master encryption key is changed
- When the database is backed up
- When the keystore passphrase is changed

Point

Do not overwrite an old keystore when backing up a keystore. This is because during database recovery, you must restore the keystore to its state at the time of database backup. When the backup data of the database is no longer required, delete the corresponding keystore.

Example

- Back up the database and the keystore on May 1, 2015.

```
> pgx_dmpall -D D:\database\inst1
> copy C:\key\store\location\keystore.ks C:\keybackup\keystore_20150501.ks
```

  Specify the following in the pgx_dmpall command:

  - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

- Change the master encryption key, and back up the keystore on May 5, 2015.

```
> psql -c "SELECT pgx_set_master_key('passphrase')" postgres
> copy C:\key\store\location\keystore.ks C:\keybackup\keystore_20150505.ks
```

  Specify the following in the psql command:

  - Specify the SQL function that sets the master encryption key in the -c option.
  - Specify the name of the database to be connected to as the argument.

If the keystore is corrupted or lost, restore the keystore containing the latest master encryption key. If there is no keystore containing the latest master encryption key, restore the keystore to its state at the time of database backup, and recover the database from the database backup. This action recovers the keystore to its latest state.

Example

- Restore the keystore containing the latest master encryption key as of May 5, 2015.

```
> copy C:\keybackup\keystore_20150505.ks C:\key\store\location\keystore.ks
```

- If there is no backup of the keystore containing the latest master encryption key, recover the keystore by restoring the keystore that was backed up along with the database on 1 May 2015.

```
> copy C:\keybackup\keystore_20150501.ks C:\key\store\location\keystore.ks
> pgx_rcvall -B E:\backup\inst1 -D D:\database\inst1 --keystore-passphrase
```

  Specify the following in the pgx_rcvall command:

  - Specify the data storage directory in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
  - Specify the backup data storage directory in the -B option.
  - The --keystore-passphrase option prompts you to enter the passphrase to open the keystore.

If you have restored the keystore, repeat the process of enabling automatic opening of the keystore. This ensures that the contents of the automatically opening keystore (keystore.aks) are identical to the contents of the restored keystore.

It is recommended that you do not back up the automatically opening keystore file, keystore.aks. If the database backup medium and the backup medium storing the automatically opening keystore are both stolen, the attacker will be able to read the data even without knowing the passphrase.

If the automatically opening keystore is corrupted or lost, you must again enable automatic opening. The keystore.aks file will be recreated from keystore.ks at this time.

See

Refer to "pgx_rcvall" and "pgx_dmpall" in the Reference for information on the pgx_rcvall and pgx_dmpall commands.

Refer to "psql" under "Reference" in the PostgreSQL Documentation for information on the psql command.

Refer to "B.2 Transparent Data Encryption Control Functions" for information on the pgx_set_master_key function.

Refer to "5.6.3 Enabling Automatic Opening of the Keystore" for information on how to enable automatic opening of the keystore.

5.7 Backing Up and Restoring/Recovering the Database

Enterprise Postgres enables you to use the five backup and recovery methods described below. Regardless of the method you use, you must back up the keystore at the same time.

Note that you must store the database and the keystore on separate data storage media. Storing both on the same data storage medium risks the danger of the encrypted data being deciphered if the medium is stolen.

Backup and recovery using WebAdmin

- Backup

  WebAdmin backs up encrypted data.

  Back up the key store after backing up the database.

- Recovery

  Restore the keystore to its state at the time of database backup. Refer to "5.6.4 Backing Up and Recovering the Keystore" for details.

  Enable automatic opening of the keystore in accordance with the procedure described in "5.6.3 Enabling Automatic Opening of the Keystore". Then, use WebAdmin to recover the database.

Backup and recovery using the pgx_dmpall and pgx_rcvall commands

- Backup

  The pgx_dmpall command backs up the encrypted data.

  Back up the key store after backing up the database.

- Recovery

  Restore the keystore to its state at the time of the database backup.

  Configure automatic opening of the key store as necessary.

  If automatic opening of the keystore is not enabled, execute the pgx_rcvall command with the --keystore-passphrase option specified. This will display the prompt for the passphrase to be entered.

Example

- Back up the database and the keystore on May 1, 2015.

```
> pgx_dmpall -D D:\database\inst1
> copy C:\key\store\location\keystore.ks C:\keybackup\keystore_20150501.ks
```

  Specify the following in the pgx_dmpall command:

  - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

- Recover the database and the keystore from the backup taken on May 1, 2015.

```
> copy C:\keybackup\keystore_20150501.ks C:\key\store\location\keystore.ks
> pgx_keystore --enable-auto-open C:\key\store\location\keystore.ks (Execute only when enabling
automatic opening)
> pgx_rcvall -B E:\backup\inst1 -D D:\database\inst1 --keystore-passphrase
```

  Specify the following in the pgx_rcvall command:

  - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
  - Specify the backup data storage directory in the -B option.
  - The --keystore-passphrase option prompts you to enter the passphrase to open the keystore.

Dump and restore using SQL

- Backup

  The files output by the pg_dump and pg_dumpall commands are not encrypted. You should, therefore, encrypt the files using OpenSSL commands or other means before saving them, as described in "5.8 Importing and Exporting the Database" below.

  Back up the key store after backing up the database.

- Restore

  If the backup data has been encrypted using, for example Open SSL commands, decrypt that data.

  The data generated by the pg_dumpall command includes a specification to encrypt tablespaces by For this reason, the pg_restore command encrypts tablespaces during restoration.

File system level backup and restore

- Backup

  Stop the instance and backup the data directory and the tablespace directory using the file copy command of the operating system. The files of encrypted tablespaces are backed up in the encrypted state.

  Back up the key store after performing the backup.

- Restore

  Restore the keystore to its state at the time of the database backup.

  Stop the instance and restore the data directory and the tablespace directory using the file copy command of the operating system.

Continuous archiving and point-in-time recovery

- Backup

  The pg_basebackup command backs up the encrypted data as is.

  Back up the key store after performing the backup.

- Recovery

  Restore the keystore to its state at the time of the database backup.

  Configure automatic opening of the key store as necessary.

  If automatic opening of the keystore is not enabled, refer to "5.3 Opening the Keystore" for information on starting an instance by specifying pg_ctl --keystore-passphrase start.

See

- Refer to "Reference" in the PostgreSQL Documentation for information on the following commands:
  - psql
  - pg_dump
  - pg_restore
  - pg_basebackup
- Refer to the Reference for information on the following commands:
  - pgx_rcvall
  - pgx_dmpall
  - pg_dumpall

If you have restored the keystore, repeat the process of enabling automatic opening of the keystore This ensures that the contents of the automatically opening keystore (keystore.aks) are identical to the contents of the restored keystore.

Refer to "5.6.3 Enabling Automatic Opening of the Keystore" for information on how to enable automatic opening of the keystore.

5.8 Importing and Exporting the Database

The files output by the COPY TO command are not encrypted. Therefore, when transferring files to other systems, you should encrypt files using OpenSSL commands, or use file transfer software that performs encrypted communication for Windows, to encrypt the data being transferred.

Use a safe method to delete obsolete plain text files.

You can use the following methods to safely delete files:

- fsutil command

 Example

```
# Export the contents of the table my_table to a CSV file.
> psql -c "COPY my_table TO 'C:\WINDOWS\Temp\my_table.csv' (FORMAT CSV)" postgres
# Encrypt the exported file.
> C:\OpenSSL-Win32\bin\openssl enc -e -aes256 -in C:\WINDOWS\Temp\my_table.csv -out my_table.csv.enc
(The user is prompted to enter the passphrase to be used for encryption)

# Check the size of plain text files, and delete them after zero padding
> dir C:\WINDOWS\Temp\my_table.csv
> fsutil file setzerodata offset=0 length=7 C:\WINDOWS\Temp\my_table.csv
> del C:\WINDOWS\Temp\my_table.csv

# Decrypt the encrypted files on other systems.
> C:\OpenSSL-Win32\bin\openssl enc -d -aes256 -in my_table.csv.enc -out my_table.csv
(The user is prompted to enter the passphrase to be used for decryption)
```

If you use COPY FROM to import data to tables and indexes in an encrypted tablespace, the imported data is automatically encrypted before being stored.

5.9 Encrypting Existing Data

You cannot encrypt existing unencrypted tablespaces. In addition, you cannot change encrypted tablespaces so that they do not encrypt.

As an alternative, transfer the tables and indexes to other tablespaces. You can use the following SQL commands for this.

```
ALTER TABLE table_name SET TABLESPACE new_tablespace;
ALTER INDEX index_name SET TABLESPACE new_tablespace;
ALTER DATABASE database_name SET TABLESPACE new_tablespace;
```

 See

Refer to "SQL Commands" under "Reference" in the PostgreSQL Documentation for information on SQL commands.

5.10 Operations in Cluster Systems

This section describes how to use transparent data encryption on cluster systems such as high-availability systems, streaming replication, C, and the Mirroring Controller option.

5.10.1 HA Clusters that do not Use Database Multiplexing

Take the following points into account when using transparent data encryption in an HA cluster environment that does not use database multiplexing.

Placement and automatic opening of the keystore file

There are two alternatives for placing the keystore file:

- Sharing the keystore file
- Placing a copy of the keystore file

Sharing the keystore file

This involves using the same keystore file on the primary server and the standby server.

As the standby server is not active while the primary server is running, this file would not be accessed simultaneously, and therefore, it can be shared.

To manage the keystore file in a more secure manner, place it on the key management server or the key management storage isolated in a secure location.

Enable the automatic opening of the keystore on both the primary and standby servers.

Placing a copy of the keystore file

This involves placing a copy of the primary server keystore file on the standby server.

You can do this if you cannot prepare a shared server or disk device that can be accessed from both the primary and standby servers.

However, if you change the master encryption key and the passphrase on the primary server, you must copy the keystore file to the standby server again.

To manage the keystore file in a more secure manner, prepare the key management server or the key management storage isolated in a secure location for both the primary and standby servers, and place the keystore files there.

Enable the automatic opening of the keystore on both the primary and standby servers. Note that copying the automatically opening keystore file (keystore.aks) to the standby server does not enable the automatic opening of the keystore.

See

Refer to the Cluster Operation Guide for information on building a cluster system environment using failover operation.

5.10.2 Database Multiplexing Mode

Note the following when using transparent data encryption in environments that use streaming replication, database multiplexing with streaming replication, or the Mirroring Controller option.

Placing the keystore file

Place a copy of the primary server keystore file on the standby server.

This is required as the keystore file cannot be shared, and both servers may need to access it simultaneously.

Point

To manage the keystore file in a more secure manner, place it on the key management server or the key management storage isolated in a secure location. A keystore used by both the primary and standby servers can be managed on the same key management server or key management storage.

However, create different directories for the keystores to be used by the primary server and the standby server. Then copy the keystore for the primary server to the directory used on the standby server.

Automatically opening the keystore

You must enable automatic opening of the keystore.

To do this, enable automatic opening of the keystore in all servers that make up database multiplexing. The settings for automatic opening of the keystore include information unique to each server, so simply copying the file does not enable it.

Changing the passphrase

Changes to the passphrase are reflected in all servers that make up database multiplexing, so no special operation is required.

Building and starting a standby server

Before using the pg_basebackup command or pgx_rcvall command to build a standby server, copy the keystore file from the primary server to the standby server. When using an automatically opening keystore, use the copied keystore file to enable automatic opening on the standby server.

Open the keystore each time you start the standby server. This step is necessary for decrypting and restoring encrypted WAL received from the primary server. To open the keystore, specify the --keystore-passphrase option in the pg_ctl command or pgx_rcvall command and enter the passphrase, or use an automatically opening keystore.

If specifying --keystore-passphrase in the pg_ct command, refer to "5.3 Opening the Keystore" for details.

Changing the master encryption key and the passphrase

Change the master encryption key and the passphrase on the primary server. You need not copy the keystore from the primary server to the standby server. You need not even restart the standby server or reopen the keystore. Changes to the master encryption key and the passphrase are reflected in the keystore on the standby server.

See

Refer to "pgx_rcvall " in the Reference for information on pgx_rcvall command.

Refer to "pg_basebackup" under "Reference" in the PostgreSQL Documentation for information on pg_basebackup command.

Refer to "High Availability, Load Balancing, and Replication" under "Server Administration" in the PostgreSQL Documentation for information on how to set up streaming replication.

5.11 Security-Related Notes

- Decrypted data is cached in the database server memory (shared buffer). As a result, unencrypted data is stored in a minidump, which is the process memory dump. You should, therefore, safely delete the memory dump. You can safely delete files by using the following command:
  - fsutil command
- Unencrypted data may be written from the database server memory to the operating system's swap area. To prevent leakage of information from the swap area, consider either disabling the use of swap area or encrypting the swap area using a full-disk encryption product.
- The content of the server log file is not encrypted. Therefore, in some cases the value of a constant specified in a SQL statement is output to the server log file. To prevent this, consider setting a parameter such as log_min_error_statement.
- When executing an SQL function that opens the keystore and modifies the master encryption key, ensure that the SQL statement containing the passphrase is not output to the server log file. To prevent this, consider setting a parameter such as log_min_error_statement. If you are executing this type of SQL function on a different computer from the database server, encrypt the communication between the client and the database server with SSL.

5.12 Tips for Installing Built Applications

With transparent data encryption, you can easily encrypt all the data in an application without modifying the application. Database administrators install built applications in the following manner. However, this procedure stores data to the default tablespace, so take necessary action if processing differs from the original design.

1. (Normal procedure) Create an owner and a database for the built application.

```
CREATE USER crm_admin ...;
CREATE DATABASE crm_db ...;
```

2. (Procedure for encryption) Create an encrypted tablespace to store the data for the built application.

```
SET tablespace_encryption_algorithm = 'AES256';
CREATE TABLESPACE crm_tablespace LOCATION 'C:\crm\data';
```

3. (Procedure for encryption) Configure an encrypted tablespace as the default tablespace for the owner of the built application.

```
ALTER USER crm_admin SET default_tablespace = 'crm_tablespace';
ALTER USER crm_admin SET temp_tablespaces = 'crm_tablespace';
```

4. (Normal procedure) Install the built application. The application installer prompts you to enter the host name and the port number of the database server, the user name, and the database name. The installer uses the entered information to connect to the database server and execute the SQL script. For applications that do not have an installer, the database administrator must manually execute the SQL script.

Normally, the application's SQL script includes logic definition SQL statements, such as CREATE TABLE, CREATE INDEX, and GRANT or REVOKE, converted from the entity-relationship diagram. It does not include SQL statements that create databases, users, and tablespaces. Configuring the default tablespace of the users who will execute the SQL script deploys the objects generated by the SQL script to the tablespace.

Chapter 6 Periodic Operations

This chapter describes the operations that must be performed periodically when running daily database jobs.

6.1 Configuring and Monitoring the Log

Enterprise Postgres enables you to output database errors and warnings to a log file.

This information is useful for identifying if errors have occurred and the causes of those errors.

By default, this information is output to the event log. It is recommended that you configure Enterprise Postgres to collect logs from its log files (for example, log_destination) before operating Enterprise Postgres.

Periodically monitor the log files to check if any errors have occurred.

See

- Refer to "Error Reporting and Logging" under "Server Administration" in the PostgreSQL Documentation for information on logs.
- Refer to "Configuring Parameters" in the Installation and Setup Guide for Server for information on log settings when operating with WebAdmin.

6.2 Monitoring Disk Usage and Securing Free Space

When a database is used for an extended period, free space on the disk is continuously consumed and in some cases the disk space runs out. When this happens, database jobs may stop and no longer run.

You should, therefore, periodically monitor the usage of disk space, and delete obsolete files located in the disk.

Monitor the disk usage of the disk where the following directories are located:

- Data storage destination directory
- Transaction log storage destination (if the transaction log is stored in a different directory from the data storage destination directory)
- Backup data storage destination directory
- Tablespace storage destination directory

6.2.1 Monitoring Disk Usage

To check the disk usage, use the following operating system commands:

- fsutil volume diskfree command

You can even use SQL statements to check tables and indexes individually.

Refer to "Determining Disk Usage" under "Server Administration" in the PostgreSQL Documentation for information on this method.

Information

If you are using WebAdmin for operations, a warning is displayed when disk usage reaches 80%

6.2.2 Securing Free Disk Space

Secure free disk space by using the following operating system commands to delete unnecessary files, other than the database, from the same disk unit.

- del command

You can also secure disk space by performing the following tasks periodically:

- To secure space on the data storage destination disk:

  Execute the REINDEX statement. Refer to "6.5 Reorganizing Indexes" for details.

- To secure space on the backup data storage destination disk:

  Execute backup using WebAdmin or the pgx_dmpall command.

6.3 Automatically Closing Connections

If an application stops responding and abnormally terminates for any reason, the connection from the application may remain active on the database server. If this situation continues for an extended period, other applications attempting to connect to the database server may encounter an error, or an error indicating that the tables are unavailable may occur.

It is, therefore, recommended that idle connections be closed automatically at regular intervals.

Set the following parameters in the postgresql.conf file to indicate the time permitted to elapse before a connection is closed.

Parameter name	Setting	Description
tcp_keepalives_idle	Time until keepalive is sent (seconds) If 0, the default value of the system is used.	Sends keepalive to an idle connection at the specified interval in seconds It is recommended to specify 30 seconds.
tcp_keepalives_interval	keepalive send interval (seconds) If 0, the default value of the system is used.	Sends keepalive at the specified interval It is recommended to specify 6 seconds.

Note

The maximum number of connections allowed is 125, unless the desktop heap setting is changed.

See

Refer to "Connection Settings" under "Server Administration" in the PostgreSQL Documentation for information on the parameters.

6.4 Monitoring the Connection State of an Application

Enterprise Postgres does not immediately delete the updated or deleted data. If the VACUUM determines there are no transactions that reference the database, Enterprise Postgres collects obsolete data.

However, obsolete data is not collected if there are connections that have remained active for an extended period or connections occupying resources. In this case the database may expand, causing performance degradation.

See

Refer to "Routine Vacuuming" under "Server Administration" in the PostgreSQL Documentation for information on the VACUUM command.

In such cases, you can minimize performance degradation of the database by monitoring problematic connections.

The following two methods are supported for monitoring connections that have been in the waiting status for an extended period:

- 6.4.1 Using the View (pg_stat_activity)
- 6.4.2 Using pgAdmin

6.4.1 Using the View (pg_stat_activity)

Use the view (pg_stat_activity) to identify and monitor connections where the client has been in the waiting status for an extended period.

Example

The example below shows connections where the client has been in the waiting status for at least 60 minutes.

However, when considering continued compatibility of applications, do not reference system catalogs directly in the following SQL statements.

```
postgres=# select * from pg_stat_activity where state='idle in transaction' and current_timestamp >
cast(query_start + interval '60 minutes' as timestamp);
-[ RECORD 1 ]----+------------------------------
datid            | 13003
datname          | db01
pid              | 4638
usesysid         | 10
usename          | fsep
application_name | apl01
client_addr      | 192.33.44.15
client_hostname  |
client_port      | 27500
backend_start    | 2015-04-24 09:09:21.730641+09
xact_start       | 2015-04-24 09:09:23.858727+09
query_start      | 2015-04-24 09:09:23.858727+09
state_change     | 2015-04-24 09:09:23.858834+09
waiting          | f
state            | idle in transaction
backend_xid      |
backend_xmin     |
query            | begin;
```

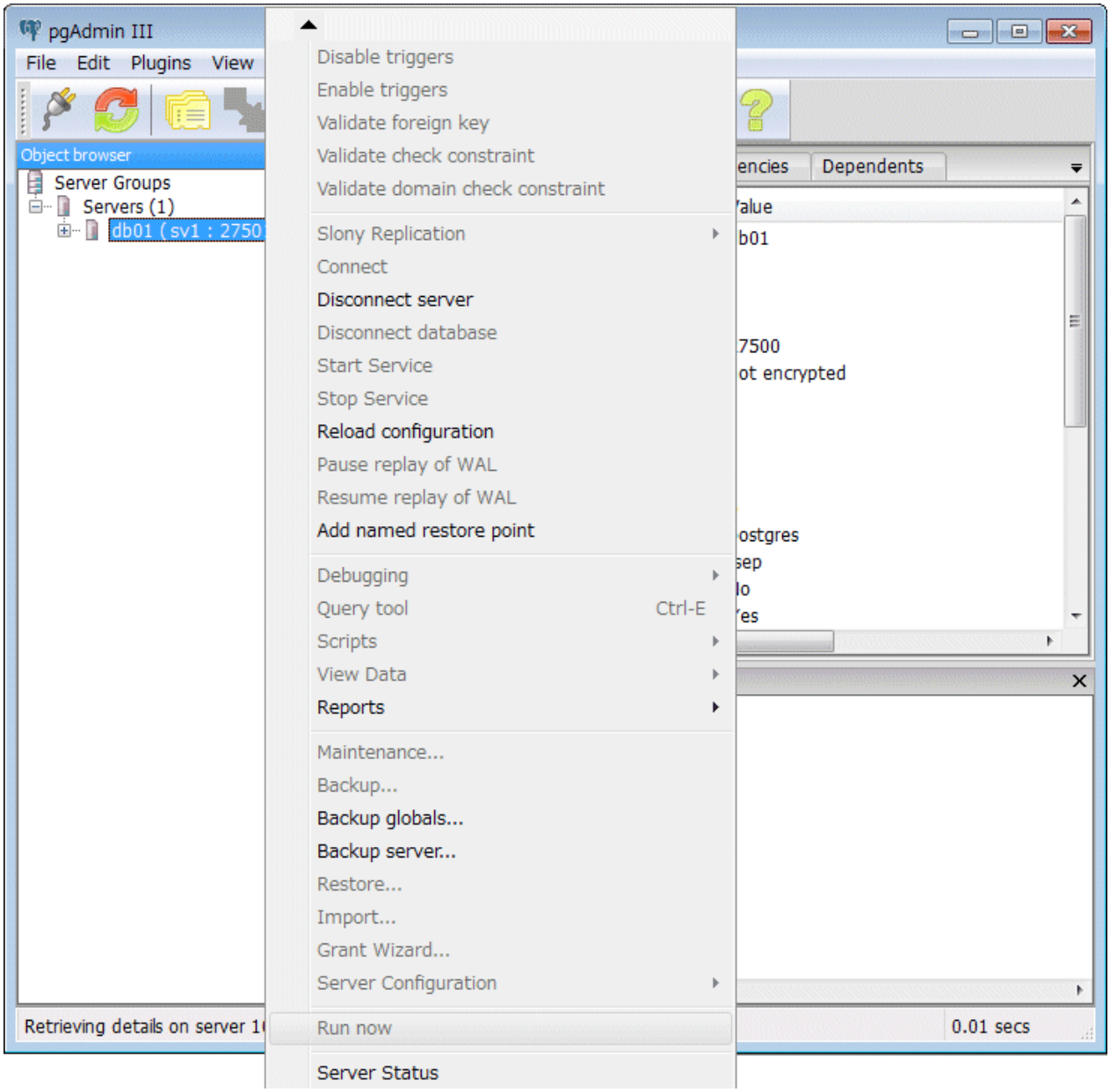
See

- Refer to "Notes on Application Compatibility" in the Application Development Guide for information on maintaining application compatibility.
- Refer to "The Statistics Collector" under "Server Administration" in the PostgreSQL Documentation for information on pg_stat_activity.

6.4.2 Using pgAdmin

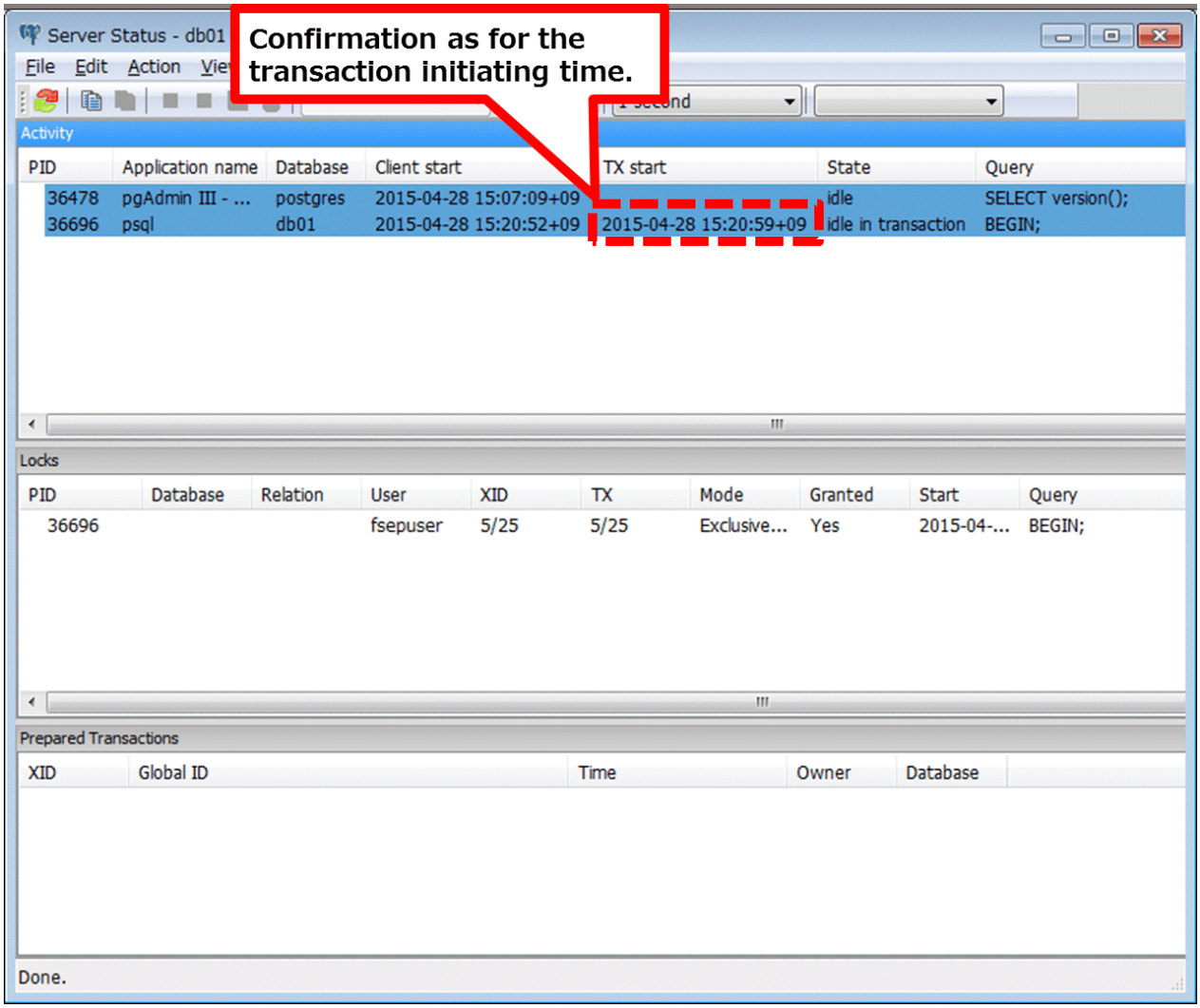
This section describes the procedure for monitoring connections using [Server Status] in pgAdmin.

1. From the [Tools] menu in pgAdmin, click [Server Status].



2. Identify client connections that have been in the waiting state for an extended period.

   From the transaction start time displayed under [TX Start], identify connections that have been in the waiting state for an extended period.



6.5 Reorganizing Indexes

Normally, a database defines indexes in tables, but if data is frequently updated, indexes can no longer use free space in the disk efficiently. This situation can also cause a gradual decline in database access performance.

To rearrange used space on the disk and prevent the database access performance from declining, it is recommended that you periodically execute the REINDEX command to reorganize indexes.

Check the disk usage of the data storage destination using the method described in "6.2 Monitoring Disk Usage and Securing Free Space".

See

Refer to "Routine Reindexing" under "Server Administration" in the PostgreSQL Documentation for information on reorganizing indexes by periodically executing the REINDEX command.

Point

Typically, reorganize indexes once a month at a suitable time such as when conducting database maintenance. Use SQL statements to check index usage. If this usage is increasing on a daily basis, adjust the frequency of recreating the index as compared to the free disk space.

The following example shows the SQL statements and the output.

However, when considering continued compatibility of applications, do not reference system catalogs and functions directly in the following SQL statements. Refer to "Notes on Application Compatibility" in the Application Development Guide for details.

[SQL statements]

```
SELECT
  nspname AS schema_name,
  relname AS index_name,
  round(100 * pg_relation_size(indexrelid) / pg_relation_size(indrelid)) / 100 AS index_ratio,
  pg_size_pretty(pg_relation_size(indexrelid)) AS index_size,
  pg_size_pretty(pg_relation_size(indrelid)) AS table_size
FROM pg_index I
  LEFT JOIN pg_class C ON (C.oid = I.indexrelid)
  LEFT JOIN pg_namespace N ON (N.oid = C.relnamespace)
WHERE
  C.relkind = 'i' AND
  pg_relation_size(indrelid) > 0
ORDER BY pg_relation_size(indexrelid) DESC, index_ratio DESC;
```

[Output]

```
 schema_name |            index_name            | index_ratio | index_size | table_size
-------------+----------------------------------+-------------+------------+------------
 public      | pgbench_accounts_pkey            |        0.16 | 2208 KB    | 13 MB
 pg_catalog  | pg_depend_depender_index         |         0.6 | 224 KB     | 368 KB
 pg_catalog  | pg_depend_reference_index        |        0.58 | 216 KB     | 368 KB
...
```

See

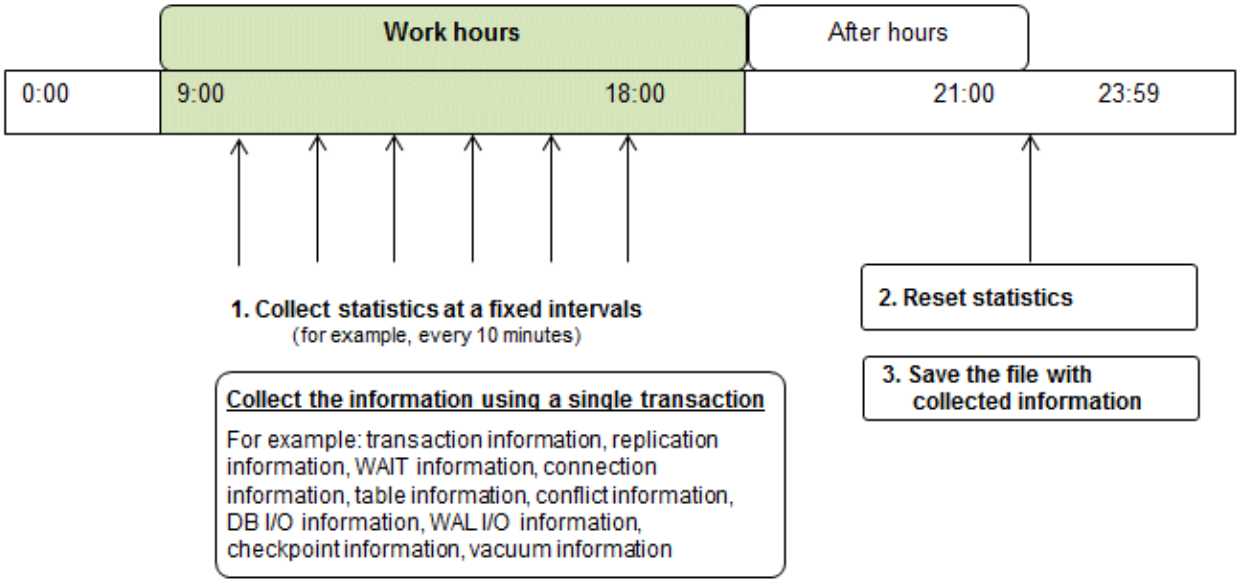
Refer to "Notes on Application Compatibility" in the Application Development Guide for information on maintaining application compatibility.

6.6 Monitoring Database Activity

Enterprise Postgres enables you to collect information related to database activity. By monitoring this information, you can check changes in the database status.

This information includes wait information for resources such as internal locks, and is useful for detecting performance bottlenecks. Furthermore, you should collect this information in case you need to request Fujitsu technical support for an investigation.

Figure 6.1 Overview of information collection



1. Collect statistics at fixed intervals during work hours.

   Accumulate the collected information into a file.

   Wherever possible, collect data from the various statistics views using a single transaction, because it enables you to take a snapshot of system performance at a given moment.

   Refer to "6.6.1 Information that can be Collected" for information on the system views that can be collected.

2. Reset statistics after work hours, that is, after jobs have finished.

   Refer to "6.6.3 Information Reset" for information on how to reset statistics.

3. Save the file with collected information.

   Keep the file with collected information for at least two days, in order to check daily changes in performance and to ensure that the information is not deleted until you have sent a query to Fujitsu technical support.

Where jobs run 24 hours a day, reset statistics and save the file with collected information when the workload is low, for example, at night.

Note

Statistics cumulatively add the daily database value, so if you do not reset them, the values will exceed the upper limit, and therefore will not provide accurate information.

The subsections below explain the following:

- Information that can be collected
- Collection configuration
- Information reset

6.6.1 Information that can be Collected

Information that can be collected is categorized into the following two types:

- Information common to PostgreSQL
- Information added by Enterprise Postgres

Information common to PostgreSQL

See

Refer to "Monitoring Database Activity" under "Server Administration" in the PostgreSQL Documentation for information on information common to PostgreSQL.

Information added by Enterprise Postgres

You can collect the following information added by Enterprise Postgres.

Table 6.1 Information added by Enterprise Postgres

View name	Description
pgx_stat_lwlock	Displays statistic related to lightweight lock, with each type of content displayed on a separate line. This information helps to detect bottlenecks. Refer to "C.2 pgx_stat_lwlock" for details.
pgx_stat_latch	Displays statistics related latches, with each type of wait information within Enterprise Postgres displayed on a separate line. This information helps to detect bottlenecks. Refer to "C.3 pgx_stat_latch" for details.
pgx_stat_walwriter	Displays statistics related to WAL writing, in a single line. Refer to "C.4 pgx_stat_walwriter" for details.
pgx_stat_sql	Displays statistics related to SQL statement executions, with each type of SQL statement displayed on a separate line. Refer to "C.5 pgx_stat_sql" for details.

6.6.2 Collection Configuration

The procedure for configuring collection depends on the information content.

- Information common to PostgreSQL
- Information added by Enterprise Postgres

Information common to PostgreSQL

See

Refer to "The Statistics Collector" in "Monitoring Database Activity" under "Server Administration" in the PostgreSQL Documentation for information on information common to PostgreSQL.

Information added by Enterprise Postgres

Information added by Enterprise Postgres is collected by default.

To enable or disable information collection, change the configuration parameters in postgresql.conf. The following table lists the views for which you can enable or disable information collection, and the configuration parameters.

View name	Parameter
pgx_stat_lwlock pgx_stat_latch	track_waits
pgx_stat_sql	track_sql

Remarks: You cannot change the collection status for pgx_stat_walwriter.

Refer to "Appendix A Parameters" for information on the parameters.

6.6.3 Information Reset

This section describes how to reset information.

Information added by Enterprise Postgres

You can reset information added by Enterprise Postgres by using the pg_stat_reset_shared function in the same way as for information common to PostgreSQL.

Configure the following parameters in the pg_stat_reset_shared function:

Function	Type of return value	Description
pg_stat_reset_shared(text)	void	Reset some cluster-wide statistics counters to zero, depending on the argument (requires superuser privileges). Calling pg_stat_reset_shared('lwlock') will zero all counters shown in pgx_stat_lwlock. Similarly, in the following cases, all values of the pertinent statistics counter are reset: - If pg_stat_reset_shared('latch') is called: All values displayed in pgx_stat_latch - If pg_stat_reset_shared('walwriter') is called: All values displayed in pgx_stat_walwriter - If pg_stat_reset_shared('sql') is called: All values displayed in pgx_stat_sql

See

Refer to "Statistics Functions" in "Monitoring Database Activity" under "Server Administration" in the PostgreSQL Documentation for information on other parameters of the pg_stat_reset_shared function.

Chapter 7 Setting up and Operating PL/extJava

This chapter provides an overview of PL/extJava, and explains how to set up and operate PL/extJava.

7.1 Overview of PL/extJava

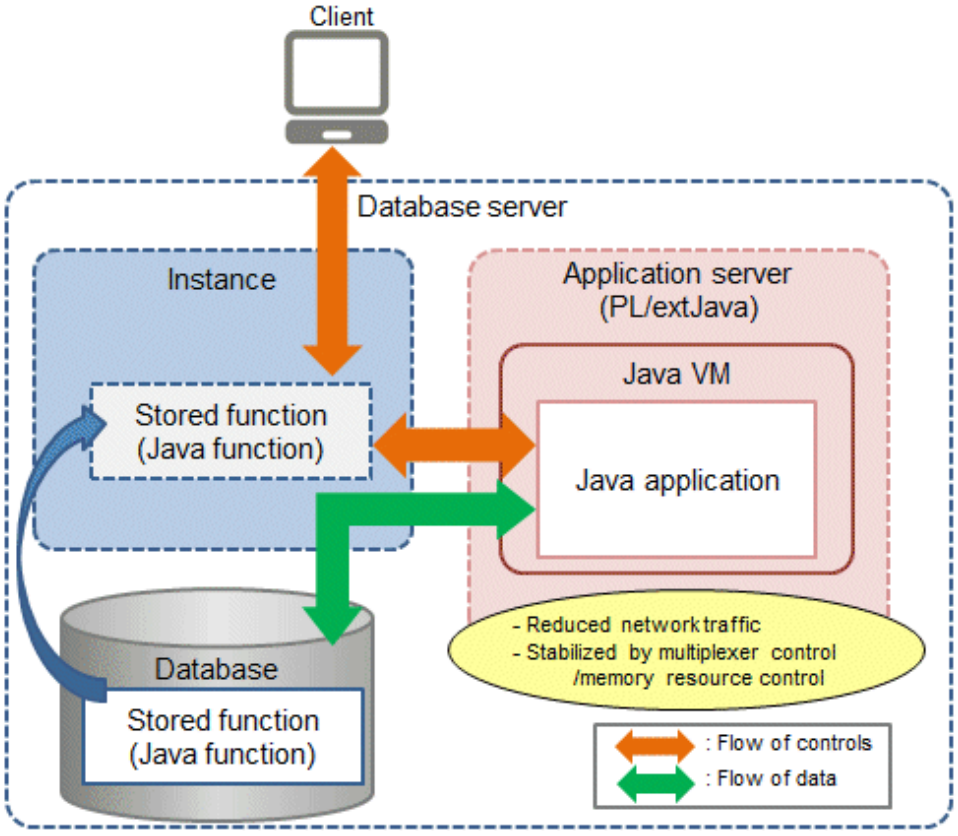
PL/extJava is a framework for incorporating the application server into the database server and controlling the Java Virtual Machine (Java VM).

Jobs where a client frequently accesses a database using SQL mean a high volume of network traffic between them. As a result, not only these jobs may experience a drop in response, but other job systems may as well. It is possible to reduce network traffic and improve the job processing time by performing business logic, implemented via stored functions, on the database server instead of on the client.

However, if several clients access the database server simultaneously, Java VM multiplexer control and memory resources must be taken into account on the database server if Java VM is run.

PL/extJava can control the Java VM multiplexer control and memory resources using the application server, so stored functions can be executed efficiently even in operation modes that have simultaneous access from many clients.

Figure 7.1 Overview of PL/extJava



Point

In this chapter, PL/extJava stored functions that are registered in the database are referred to as "Java functions". Applications that operate on the Java VM are referred to as "Java applications".

Note

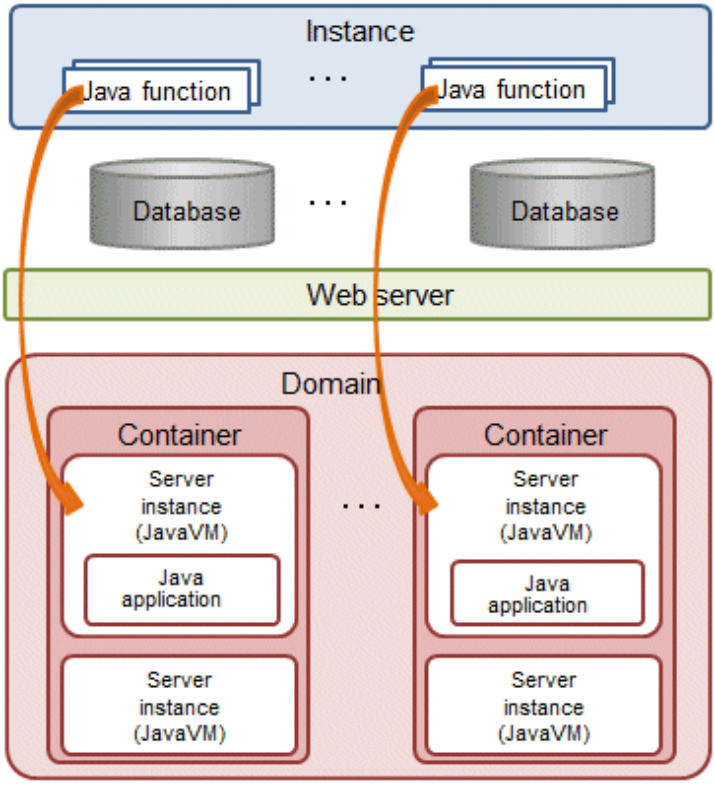
- Java functions run on different connections from the ones between the client and the database server. Jobs being migrated to Java functions must therefore be run as independent transactions.

- Names of databases that use PL/extJava must be specified in up to 28 bytes. Names of roles for connecting from the application server to the database must be specified in up to 8 bytes.

7.1.1 PL/extJava Configuration

This section explains the configuration of PL/extJava.

Figure 7.2 PL/extJava configuration



The elements in the figure are explained below.

- Instance

  An instance of Enterprise Postgres.

  If a Java function is called from the client, the web server is requested to execute the Java application.

  Refer to "7.2.4 Registering Java Functions" for details.

- Web server

  The web server receives the request from the Java function to execute the Java application.

  Based on this request, a server instance (Java VM) in the container is requested to execute the Java application.

  If there are multiple server instances (Java VM) in the container, the request will be allocated to the optimal server.

- Container

  The container is the execution environment for the Java application.

  Each container corresponds to one database of an instance that executes a Java function.

- Server instance (Java VM)

  A server instance (Java VM) is a single Java VM. Java applications can be executed simultaneously on a server instance (Java VM).

  The maximum number of Java applications that can be executed simultaneously can be extended by adding server instances (Java VM).

- Domain

  The domain centrally manages the containers.

  Each domain corresponds to a single instance. The directory that stores the domain resources is known as the "domain root".

7.1.2 Application Servers

An application server based on Interstage Application Server technology, which is incorporated in Enterprise Postgres, can be used to achieve the benefits listed below.

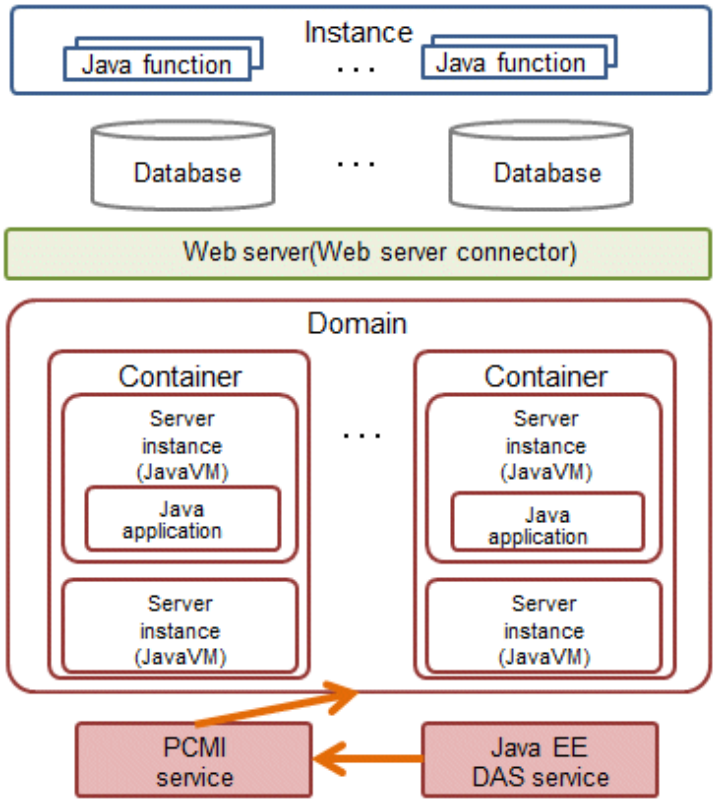
(Hereafter, an application server based on Interstage Application Server technology, which is incorporated in Enterprise Postgres, is referred to as an "Enterprise Postgres Java application server").

- The pgx_jadmin command can be used. This command simplifies the settings and controls for the Enterprise Postgres Java application server, enabling the user to easily perform setup and tuning, even without a detailed understanding of Enterprise Postgres Java application server.
- Features of Enterprise Postgres Java application server such as those listed below can be used transparently to easily achieve stable operation.
  - If several Java functions are executed, Java applications will be executed on the optimized number of Java VMs, so operations can be performed with the minimum required memory resources.
  - In anticipation of Java full garbage collection, the relevant Java VM can be disconnected to prevent any unexpected delay of Java applications.
  - Java VM heap usage and garbage collection frequency can be monitored, and risk indicators can be reported in alert notifications. (Predictive monitoring feature)
  - Java VM heartbeat monitoring enables the logging of abnormal operations of Java applications, which facilitates the efficient investigation of error causes. (Timer feature)

Refer to "pgx_jadmin" in the Reference for information on the pgx_jadmin command.

If using an Enterprise Postgres Java application server, the framework will be configured as shown below.

Figure 7.3 Configuration when using Enterprise Postgres Java application server



The specific elements of Enterprise Postgres Java application server shown in the figure are explained below.

- Web server connector

  The role of the web server connector is to transfer requests received by the web server to the container.

- PCMI service

  This service manages (starts, stops, and monitors) the Java EE DAS service and server instances (Java VM).

- Java EE DAS service

  This service integrates with the PCMI service to manage container operations.

  It receives operation requests from the pgx_jadmin command.

Note

Note the following points when using Enterprise Postgres Java application server:

- Only one domain can be created per machine. Therefore only one database cluster that uses PL/extJava can be created per machine.
- Up to 64 Java applications can be executed simultaneously by a single server instance (Java VM). To simultaneously execute more applications than this, use the pgx_jadmin command to add server instances (Java VM) as required.

7.1.3 User Definitions

This section explains the users required by PL/extJava.

- Operating system administrator user (the operating system default is "Administrator")

  The operating system administrator user becomes the domain administrator. Furthermore, the operating system administrator user also becomes the effective user of processes that configure the application server container, so file access via Java applications, for example, is performed with operating system administrator user privileges.

  If using the features of Enterprise Postgres Java application server incorporated into Enterprise Postgres:

- This is the effective user of containers and the pgx_jadmin command.
- Specify this user in the "--dbadminuser" option of "pgx_jadmin init-domain".

- Users who connect to the database

  If Java applications access the database, use the JDBC connection pool. These are the authenticated users for this access.

  Use the CREATE USER statement to add users who connect to the database.

  Refer to "7.2.3.1 Configuring Database Clusters" if using a Enterprise Postgres Java application server.

7.2 Setting up PL/extJava

This section explains the setting up of PL/extJava.

Perform the procedure below to set up PL/extJava if using Enterprise Postgres Java application server.

Note

- PL/extJava cannot be set up using WebAdmin or pgAdmin.
- Only operations in the IPv6/IPv4 dual stack environment are supported. Operations are not supported if IPv4 is disabled.
- Java functions cannot be called if the Enterprise Postgres installation directory name is longer than 50 bytes. If the error message 16551 is notified and the HTTP status code 404 is returned, check the error messages in the event log. If error message IJServer12042 has been output, ensure that the length of the Enterprise Postgres installation directory name is no longer than 50 bytes and reinstall.

Setup flow

Perform the following procedure to set up PL/extJava.

Create a database cluster in advance.

1. Preparing Port Numbers
2. Creating Domains
3. Creating PL/extJava
4. Registering Java Functions

7.2.1 Preparing Port Numbers

This section explains the port numbers used in PL/extJava.

Port numbers for domain management

Usage	Default port number	Specification method	Description
Port number for domain management	27530	pgx_jadmin init-domain or pgx_jadmin modify-domain-port	Port numbers for domain management. Three port numbers are required.
	27521		
	27522		

See

Refer to "7.2.2 Creating Domains" for information on pgx_jadmin init-domain.

Refer to "7.3.3.3 Changing Port Numbers" for information on pgx_jadmin modify-domain-port.

Port numbers for server instance (Java VM) management

The following port numbers are required for each server instance (Java VM).

Usage	Default port number	Specification method	Description
Port number for server instance (Java VM) management	27531 27532	pgx_jadmin create-container --instanceport, pgx_jadmin add-instance --instanceport, or pgx_jadmin modify-instance-port --instanceport	Used for operating a server instance (Java VM). Two port numbers must be configured for each server instance (Java VM).

See

Refer to "7.2.3.2 Creating Containers" for information on pgx_jadmin create-container.

Refer to "7.3.3.1 Adding or Deleting Server Instances (Java VM)" for information on pgx_jadmin add-instance.

Refer to "7.3.3.3 Changing Port Numbers" for information on pgx_jadmin modify-domain-port.

Note

- For the port number, specify an unused port number in the following range:
  - Windows Server(R) 2008, Windows Server(R) 2008 R2, Windows Server(R) 2012, or Windows Server(R) 2012 R2:1024 to 49151
- Configuring the firewall

  The port number specified in the pgx_jadmin command is used for communication within the local machine. Use the firewall to restrict access to each port number. This prevents unauthorized access and ensures security.

7.2.2 Creating Domains

Create a domain for using PL/extJava.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Edit the hosts file

   Add the information of the host that builds PL/extJava to the hosts file.

3. Create the domain

   Use the init-domain subcommand of the pgx_jadmin command to create the domain.

   Example

   If the domain root is "C:\domain"

```
> pgx_jadmin init-domain --shareddir C:\domain --domainport 27530,27521,27522 --pgdata c:
\database\inst1
```

See

Refer to "pgx_jadmin" in the Reference for information on the init-domain subcommand.

7.2.3 Creating PL/extJava

This section explains how to create PL/extJava.

7.2.3.1 Configuring Database Clusters

Configure the definition of the database cluster.

1. Configure the "listen_addresses" setting in the postgresql.conf file

   A loopback address is used to connect the Java application to the database. Therefore, ensure that "listen_addresses" is configured to allow connection from localhost.

   See

   Refer to "Connections and Authentication" under "Server Administration" in the PostgreSQL Documentation for details.

2. Add users who connect to the database

   If Java applications access the database, use the JDBC connection pool.

   Use the CREATE USER statement to add users who connect to the database. Specify the additional user names when creating the container.

   Example

   If the user name is "user01"

   ```
   db01=# create user user01;
   ```

3. Configure the pg_hba.conf file

   Establish a local connection to the database from the Java application. Ensure that the settings allow local connections.

Parameter	Parameter value
TYPE	"host"
DATABASE	Name of the database the Java application connects to
USER	User name used when the Java application is connected to the database
ADDRESS	127.0.0.1/32
AUTH-METHOD	Use a method other than trust authentication

   See

   Refer to "The pg_hba.conf File" under "Server Administration" in the PostgreSQL Documentation for details.

4. Configure "max_connections" in the postgresql.conf file

   Add the number of connections used when connecting from Enterprise Postgres Java application server to the database, calculated as shown below, to the existing value of max_connections.

   ```
   numberOfJavaVmServerInstances x 64
   ```

See

Refer to "Configuring Remote Connections" in the Installation and Setup Guide for Server for details.

7.2.3.2 Creating Containers

Create one container per database within a database cluster that uses Java functions.

When you create a container, a server instance (Java VM) is created in the container at the same time. Settings for the JDBC connection pool and the data source used to access the database from the Java application are also configured at this time. The name of the server instance (Java VM) will be "*databaseName-serialNumber*". Confirm the name of the server instance (Java VM) using the list-container subcommand of the pgx_jadmin command.

See

Refer to "pgx_jadmin" in the Reference for information on the list-container subcommand.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Create the container

   Use the create-container subcommand of the pgx_jadmin command to create the container. Information required for the database connection can still be modified later.

   Example

   ```
   > pgx_jadmin create-container --dbname db01 --dbuser user01 --dbpassword password1 --
   instanceport 27531,27532
   ```

   See

   Refer to "pgx_jadmin" in the Reference for information on the create-container subcommand.

3. Add the server instances (Java VM)

   To simultaneously execute 65 or more Java applications, add the required extra number of server instances (Java VM).

   Refer to "7.3.3.1 Adding or Deleting Server Instances (Java VM)" for information on adding server instances (Java VM).

4. Start the instance again

   Refer to "2.1 Starting and Stopping an Instance" for information on how to start the instance.

   Refer to "8.10 Actions in Response to Instance Startup Failure" if the instance fails to start.

5. Install the PL/extJava extension

   Execute the CREATE EXTENSION statement and install the PL/extJava extension.

   ```
   db01=# CREATE EXTENSION plextjavau;
   ```

7.2.4 Registering Java Functions

This section explains how to register Java functions and store Java applications.

This procedure must be performed each time a Java function is created.

Registering Java Functions

1. Connect to the database

   As the instance administrator, connect to the database that corresponds to the container that was created using the psql command or pgAdmin.

2. Register the function in the database

   Java functions run on sessions that are not dependent on the caller session. To operate a Java function using the caller session, use the SET statement to set the plextjava.separate_session parameter to "off" and then register it.

```
SET plextjava.separate_session=off
```

   This section explains the CREATE FUNCTION statement used when registering a function in the database.

   Syntax:

```
CREATE  [ OR  REPLACE ]  FUNCTION
    name  ( [ IN ] argtype [, ...] )
    [ RETURNS  rettype ]
    AS  'definition'
    LANGUAGE  lang_name
```

   name

   Specify the name of the Java function.

   The name of the Java function does not need to match the name of the Java application method.

   A qualified schema name can be used. Create the schema in advance.

   argtype

   Specify the data type of the Java function argument.

   rettype

   Specify the data type of the Java function return value.

   definition

   Specify "*packageName.className. methodName*" or "*className.methodName"* of the Java application.

   Do not specify the method arguments as they will be invalid.

   lang_name

   Specify "plextjavau".

Example

If the schema name is "javatest", the Java function name is "java_addOne", and the "*packageName.className.methodName*" of the Java application is "org.postgresql.plextjava.example.Parameters.addOne"

```
db01=# CREATE FUNCTION javatest.java_addOne(INTEGER)
                  RETURNS INTEGER
                  AS 'org.postgresql.plextjava.example.Parameters.addOne'
                  LANGUAGE plextjavau;
```

See

- Refer to "Relationship between the Java Function Data Type and Application Data Type" in the Application Development Guide for information on data types.
- Refer to "CREATE FUNCTION" under "Reference" in the PostgreSQL Documentation for details.

3. Set privileges for users connecting to the database

   Grant privileges to users for database objects (such as tables, columns, and views) that will be accessed when accessing the database from the Java application. Use the GRANT statement to grant privileges.

   Example

   Grant privileges to user name "user01" to add data to table "table01"

   ```
   db01=# GRANT INSERT ON table01  to user01;
   ```

4. Set privileges for Java functions

   Grant execution privileges for Java functions to users who call Java functions. Use the GRANT statement to grant privileges.

   Note

   Grant execution privileges for a Java function only to users who have permission to access the resources accessed by that functions, otherwise the users may inadvertently be granted improper access to those resources via the function.

   Example

   Grant privileges to user "user04" for the Java function "java_addOne"

   ```
   db01=# GRANT EXECUTE ON FUNCTION javatest.java_addOne(INTEGER) to user04;
   ```

5. Disconnect from the database.

See

Refer to "Reference" in the PostgreSQL Documentation for information on the GRANT statement.

Storing Java applications

1. Copy the Java application

   Manually copy the Java application in jar format to the directory below.

   ```
   domainRoot\plextjava\databaseName
   ```

7.3 PL/extJava Operation

This section explains the operation of PL/extJava.

If using the Enterprise Postgres Java application server, operate using the procedure explained below.

7.3.1 Starting and Stopping Containers

The container will start or stop automatically when the instance is started or stopped.

7.3.2 Checking PL/extJava

This section explains how to check PL/extJava.

7.3.2.1 Checking the Domain Information

Use the list-domain subcommand of the pgx_jadmin command to check the domain information.

 Example

```
> pgx_jadmin list-domain
```

Sample display

```
domain status: running
shareddir: C:\domain
domainport: 27530,27521,2752
dbadminuser: fsepuser
datadir: C:\database\inst1
```

 See

Refer to "pgx_jadmin" in the Reference for information on the list-domain subcommand.

7.3.2.2 Checking the Container Information

Use the list-container subcommand of the pgx_jadmin command to check the container information.

 Example

If the database name is "db01"

```
> pgx_jadmin list-container --dbname db01
```

Sample display

```
container status: running
dbname: db01
instance: db01-1 27531,27532
instance: db01-2 27801,27802
dbport: 27011
dbuser: user01
```

 See

Refer to "pgx_jadmin" in the Reference for information on the list-container subcommand.

7.3.3 Changing PL/extJava

This section explains how to change PL/extJava.

7.3.3.1 Adding or Deleting Server Instances (Java VM)

This section explains how to add or delete a server instance (Java VM).

A server instance (Java VM) is a single Java VM. Up to 64 Java applications can be executed simultaneously on a single server instance (Java VM).

The number of Java applications that can be executed simultaneously can be extended by adding server instances (Java VM). To decrease the number of Java applications that are executed simultaneously, delete the server instances (Java VM) that were added.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Change the database cluster configuration

   Change the maximum number of simultaneous instance connections (max_connections).

   Refer to "7.2.3.1 Configuring Database Clusters" for information on how to change the number of simultaneous connections.

3. Add or delete the server instance (Java VM)

   Add or delete the server instance (Java VM) using the add-instance subcommand or the delete-instance subcommand of the pgx_jadmin command.

   Example

   Add a server instance (Java VM)

   ```
   > pgx_jadmin add-instance --dbname db01 --instanceport 27801,27802
   ```

   Delete a server instance (Java VM)

   ```
   > pgx_jadmin delete-instance --dbname db01
   ```

   See

   Refer to "pgx_jadmin" in the Reference for information on the add-instance and delete-instance subcommands.

4. Start the instance again

   Refer to "2.1 Starting and Stopping an Instance" for information on how to start the instance.

   Refer to "8.10 Actions in Response to Instance Startup Failure" if the instance fails to start.

7.3.3.2 Changing the Database Connection Information

This section explains how to change database connection information.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Change the database connection information

   Use the modify-container-db subcommand of the pgx_jadmin command to change the database connection information.

   You can change the following database connection information:

   - Instance port number
   - User name used for database connection
   - User password used for database connection

   Example

   If changing the name of the user connecting to the database to "user02"

   ```
   > pgx_jadmin modify-container-db --dbname db01 --dbuser user02
   ```

See

Refer to "pgx_jadmin" in the Reference for information on the modify-container-db subcommand.

3. Start the instance again

   After changing the instance connection information, start the instance.

   Refer to "2.1 Starting and Stopping an Instance" for information on how to start the instance.

   Refer to "8.10 Actions in Response to Instance Startup Failure" if the instance fails to start.

7.3.3.3 Changing Port Numbers

This section explains how to change the domain management port numbers, or the port numbers for managing a server instance (Java VM).

Changing the domain management port numbers

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Check the port numbers before they are changed

   Use the list-domain subcommand of the pgx_jadmin command to check the port numbers before they are changed.

   Example

```
> pgx_jadmin list-domain
```

   See

   Refer to "pgx_jadmin" in the Reference for information on the list-domain subcommand.

3. Change the domain management port numbers

   Use the modify-domain-port subcommand of the pgx_jadmin command to change the port numbers.

   Example

   If changing the domain management port numbers to "26600,26601,26602"

```
> pgx_jadmin modify-domain-port --domainport 27600,27601,27602
```

   See

   Refer to "pgx_jadmin" in the Reference for information on the modify-domain-port subcommand.

4. Check the port numbers after changing them

   Use the list-domain subcommand of the pgx_jadmin command to check that the port numbers were changed.

5. Start the instance again

   Refer to "2.1 Starting and Stopping an Instance" for information on how to start the instance.

Refer to "8.10 Actions in Response to Instance Startup Failure" if the instance fails to start.

Changing the server instance (Java VM) management port numbers

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Check the port numbers before they are changed

   Use the list-container subcommand of the pgx_jadmin command to check the port numbers before they are changed.

    Example

   ```
   > pgx_jadmin list-container --dbname db1
   ```

    See

   Refer to "pgx_jadmin" in the Reference for information on the list-container subcommand.

3. Change the server instance (Java VM) management port numbers

   Use the modify-instance-port subcommand of the pgx_jadmin command to change the port numbers.

    Example

   To change the server instance (Java VM) management port numbers to "26701,26702"

   ```
   > pgx_jadmin modify-instance-port --instance db1-1 --instanceport 27701,27702
   ```

    See

   Refer to "pgx_jadmin" in the Reference for information on the modify-instance-port subcommand.

4. Check the port numbers after changing them

   Use the list-container subcommand of the pgx_jadmin command to check that the port numbers were changed.

5. Start the instance again

   Refer to "2.1 Starting and Stopping an Instance" for information on how to start the instance.

   Refer to "8.10 Actions in Response to Instance Startup Failure" if the instance fails to start.

7.3.4 Deleting Java Functions

This section explains how to delete Java functions and Java applications from PL/extJava.

 Point

Java functions can be deleted while the instance is running.

Delete a Java function

1. Connect to the database

   As the instance administrator, connect to the database that corresponds to the container that was created using the psql command or pgAdmin.

2. Delete the Java function

   Delete the Java function that was registered in the database.

```
db01=# DROP FUNCTION javaFunctionName(argument);
```

3. Disconnect from the database.

See

Refer to "DROP FUNCTION" under "Reference" in the PostgreSQL Documentation for details.

Delete the Java application

1. Delete the Java application
   Delete the jar file of the following directory.

```
domainRoot\plextjava\databaseName
```

7.3.5 Deleting Containers

This section explains how to delete containers.

1. Delete the Java function

   As the instance administrator, connect to the database that corresponds to the container that was created.

   Delete all Java functions that were registered in the database.

   Refer to "7.3.4 Deleting Java Functions" for details.

2. Uninstall the PL/extJava extension

   As the instance administrator, connect to the database that corresponds to the container that was created.

   Execute the DROP EXTENSION statement and uninstall the PL/extJava extension.

```
db01=# DROP EXTENSION plextjavau;
```

3. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

4. Delete the container

   Use the delete-container subcommand of the pgx_jadmin command to delete the container.

Example

```
> pgx_jadmin delete-container --dbname db01
```

See

Refer to "pgx_jadmin" in the Reference for information on the delete-container subcommand.

7.3.6 Deleting Domains

This section explains how to delete domains.

Note

When a domain is deleted, the domain root directory is also deleted. If any files in this directory are required, such as the Java applications, perform a backup prior to deleting the domain.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Delete the domain

   Use the delete-domain subcommand of the pgx_jadmin command to delete the domain.

   ### Example

   ```
   > pgx_jadmin delete-domain
   ```

   ### See

   Refer to "pgx_jadmin" in the Reference for information on the delete-domain subcommand.

7.3.7 Backup and Restore

This section explains how to back up and restore PL/extJava,

Back up PL/extJava to guard against issues such as a failure of the disk where the PL/extJava domains and containers are placed.

For example, backup should be performed when any of the following occur:

- Modifications are made to PL/extJava (such as to its containers or server instances (Java VM))
- Java functions are added, changed, or deleted

Note

When re-creating the instance administrator after a system disk failure, ensure that it has the same user name before the failure. If the user name is different from the backup data, restoration will fail.

7.3.7.1 Backup Method

This section explains how to back up PL/extJava.

If using the failover operation, back up the active node.

If using database multiplexing mode, back up the primary server.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Prepare the backup storage disk for PL/extJava

   Prepare a disk separate from the disk used to store the PL/extJava resources. Ensure that the disk has at least twice the capacity of the domain root.

3. Back up PL/extJava

   As the operating system administrator user, back up the PL/extJava resources using the backup subcommand of the pgx_jadmin command.

   Note

   Do not specify any of the following directories as the PL/extJava backup storage directory:

   - The instance data storage destination, the backup storage destination, or the domain root
   - A directory in the instance data storage destination, the backup storage destination, or the domain root
   - A directory that places the instance data storage destination, the backup storage destination, and the domain root in a subdirectory of a directory that stores the PL/extJava backup

   Example

   If the backup storage directory for PL/extJava is "D:\backup"

   ```
   > pgx_jadmin backup --backupdir D:\backup
   ```

   See

   Refer to "pgx_jadmin" in the Reference for information on the backup subcommand.

7.3.7.2 Restore Method

This section explains how to restore PL/extJava from the point it was backed up.

1. Stop the instance

   Refer to "2.1 Starting and Stopping an Instance" for information on how to stop the instance.

   Refer to "8.11 Actions in Response to Failure to Stop an Instance" if the instance fails to stop.

2. Restore the instance

   Restore the instance if required.

3. Restore PL/extJava

   As the operating system administrator user, restore the PL/extJava resources using the restore subcommand of the pgx_jadmin command.

   Example

   If the backup storage directory for PL/extJava is "D:\backup"

   ```
   > pgx_jadmin restore --backupdir D:\backup
   ```

See

Refer to "pgx_jadmin" in the Reference for information on the restore subcommand.

Chapter 8 Actions when an Error Occurs

This chapter describes the actions to take when an error occurs in the database or an application, while Enterprise Postgres is operating.

Depending on the type of error, it may be necessary to recover the database cluster. The recovery process recovers the following resources:

- Data storage destination
- Transaction log storage destination (if the transaction log is stored in a separate disk from the data storage destination)
- Backup data storage destination

Note

Even if a disk is not defective, the same input-output error messages, as those generated when the disk is defective, may be output. The recovery actions differ for these error messages.

Check the status of the disk, and select one of the following actions:

- If the disk is defective

  Refer to "8.1 Recovering from Disk Failure (Hardware)", and take actions accordingly.

- If the disk is not defective

  Refer to "8.13 I/O Errors Other than Disk Failure", and take actions accordingly.

  A few examples of errors generated even if the disk is not defective include:

  - Network error with an external disk
  - Errors caused by power failure or mounting issues

Determining the cause of an error

If an error occurs, refer to the WebAdmin message and the event log, and determine the cause of the error.

See

Refer to "Configuring Parameters" in the Installation and Setup Guide for Server for information on server logs.

Approximate recovery time

The formulas for deriving the approximate recovery time of resources in each directory are given below.

- Data storage destination or transaction log storage destination

```
Recovery time = (usageByTheDataStorageDestination + usageByTheTransactionLogStorageDestination) /
diskWritePerformance x 1.5
```

  - *usageByTheDataStorageDestination*: Disk space used by the database cluster
  - *usageByTheTransactionLogStorageDestination*: Disk space used by the transaction log stored outside the database cluster
  - *diskWritePerformance*: Measured maximum data volume (bytes/second) that can be written per second in the system environment where the operation is performed
  - 1.5: Coefficient assuming the time excluding disk write, which is the most time-consuming step

- Backup data storage destination

```
Recovery time = usageByTheBackupDataStorageDestination / diskWritePerformance x 1.5
```

  - *usageByTheBackupDataStorageDestination*: Disk space used by the backup data

- *diskWritePerformance*: Measured maximum data volume (bytes/second) that can be written per second in the system environment where the operation is performed
- 1.5: Coefficient assuming the time excluding disk write, which is the most time-consuming step

8.1 Recovering from Disk Failure (Hardware)

This section describes how to recover database clusters to a point immediately before failure, if a hardware failure occurs in the data storage disk or the backup data storage disk.

There are two methods of recovery:

- 8.1.1 Using WebAdmin
- 8.1.2 Using Server Command

Point

Back up the database cluster after recovering it. Backup deletes obsolete archive logs (transaction logs copied to the backup data storage destination), freeing up disk space and reducing the recovery time.

8.1.1 Using WebAdmin

Recover the database cluster by following the appropriate recovery procedure below for the disk where the failure occurred.

If failure occurred in the data storage disk or the transaction log storage disk

Follow the procedure below to recover the data storage disk or the transaction log storage disk.

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance. Refer to "2.1.1 Using WebAdmin" for information on how to stop an instance. WebAdmin automatically stops instances if recovery of the database cluster is performed without stopping the instance.

3. Recover the failed disk

   Replace the disk, and then recover the volume configuration information.

4. Create a tablespace directory

   If a tablespace was defined after backup, create a directory for it.

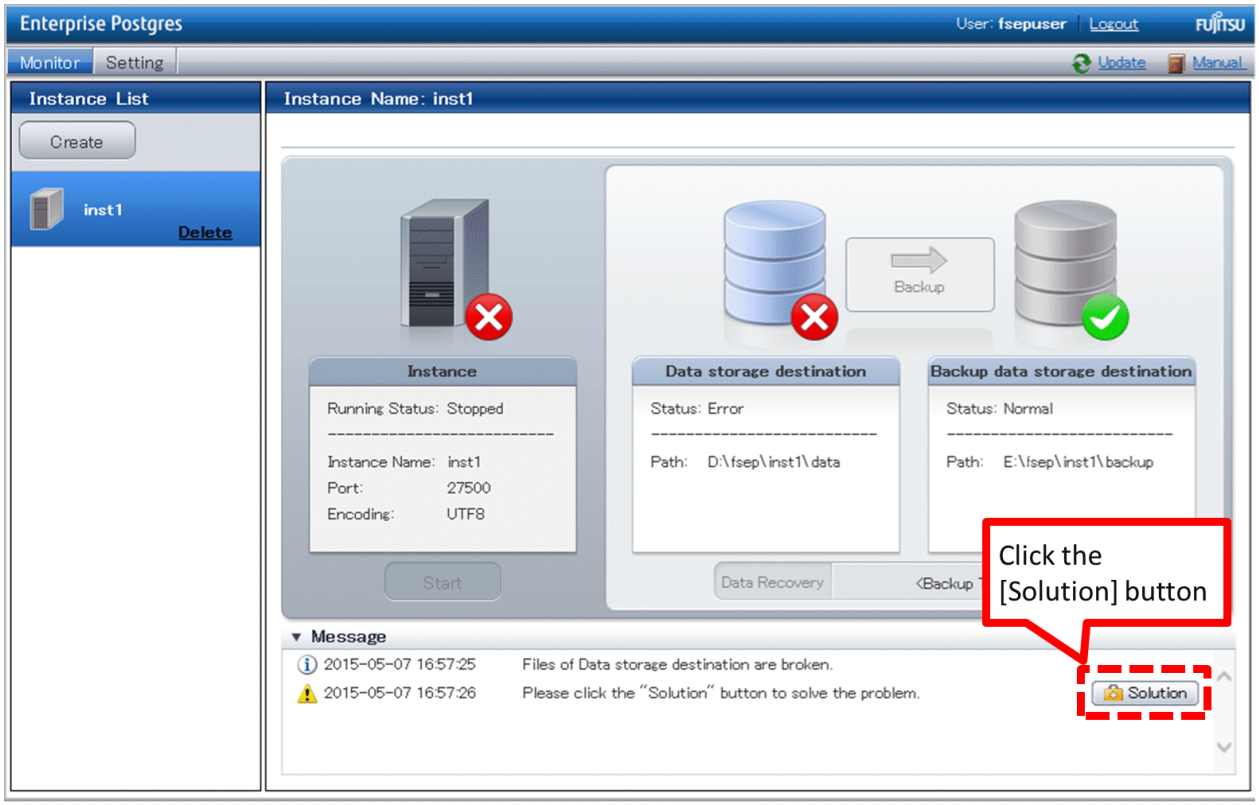
5. Recover the keystore, and enable automatic opening of the keystore

   Do the following if the data in the database has been encrypted:

   - Restore the keystore to its state at the time of the database backup.
   - Enable automatic opening of the keystore.

6. Recover the database cluster

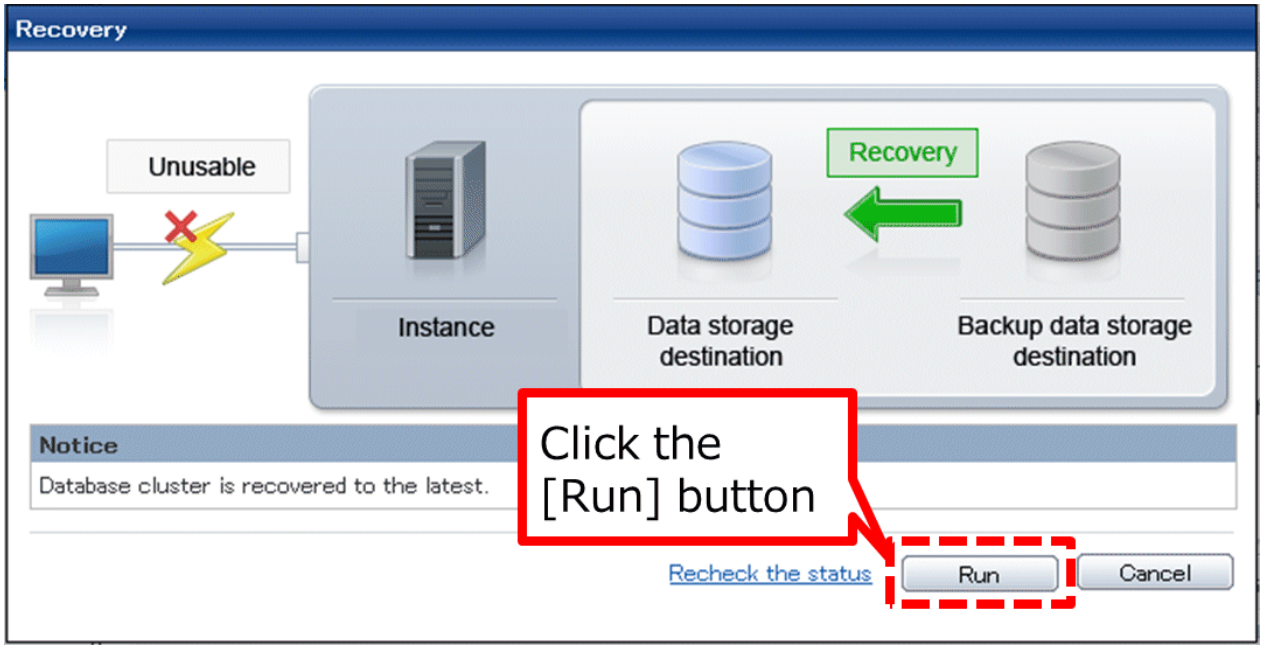
   Log in to WebAdmin, and in the [Monitor] window, click the [Solution] button for the error message.



7. Run recovery

   In the [Recovery] dialog box that appears, click [Run].

   [Recovering] is displayed in the [Monitor] window, and recovery is performed. An instance is automatically started when recovery is successful.



Note

WebAdmin does not support recovery of hash index. If you are using a hash index, then after recovery, execute the REINDEX command to rebuild it. Use of hash indexes is not recommended.

8. Resume applications

   Resume applications that are using the database.

Point

WebAdmin may be unable to detect disk errors, depending on how the error occurred.

If this happens, refer to "8.10.4 Other Errors" to perform recovery.

If failure occurred on the backup data storage disk

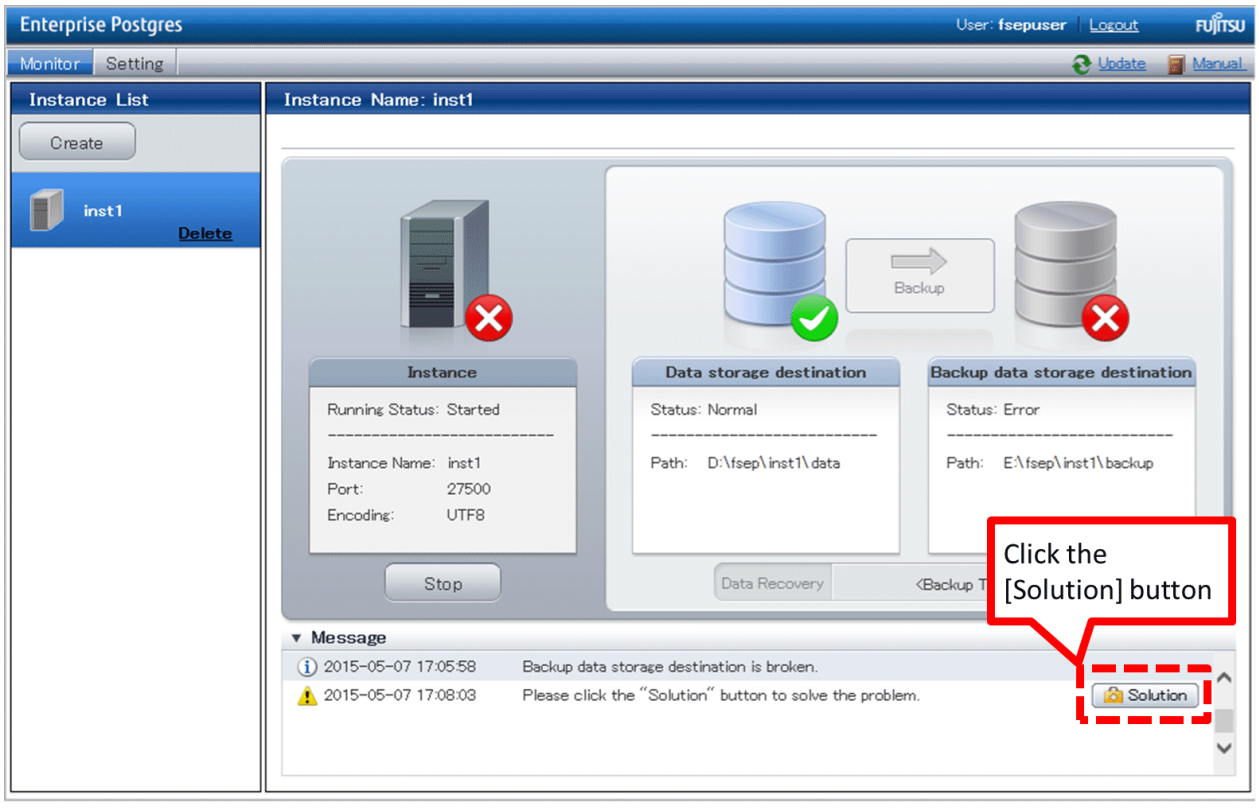
Follow the procedure below to recover the backup data storage disk.

1. Recover the failed disk

   Replace the disk, and then recover the volume configuration information.

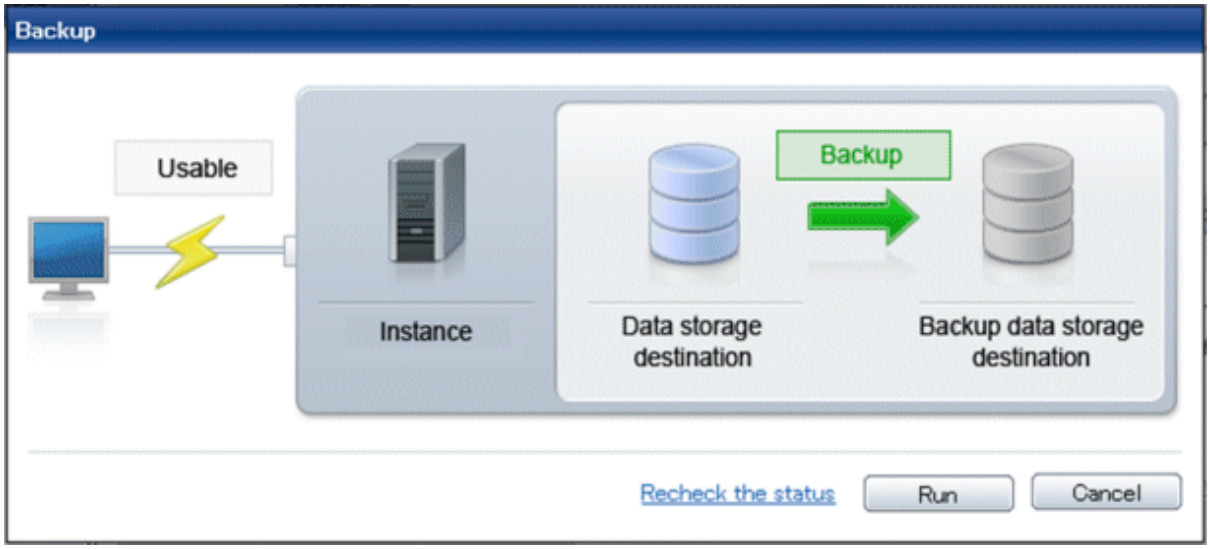
2. Recover the backup data

   Log in to WebAdmin, and in the [Monitor] window, click the [Solution] button for the error message.



3. Run backup

   Perform backup to enable recovery of the backup data. In the [Backup] dialog box that appears, click [Run]. [Backuping] is displayed in the [Monitor] window, and the backup is performed. An instance is automatically started when backup is performed.



Point

If you click [Recheck the status], the resources in the data storage destination and the backup data storage destination are reconfirmed. As a result, the following occurs:

- If an error is not detected

  The status of the data storage destination and the backup data storage destination returns to normal, and it is possible to perform operations as usual.

- If an error is detected

  An error message is displayed in the message list again. Click the [Solution] button, and resolve the problem by following the resolution for the cause of the error displayed in the dialog box.

8.1.2 Using Server Command

Recover the database cluster by following the appropriate recovery procedure below for the disk where the failure occurred.

If failure occurred on the data storage disk or the transaction log storage directory

Follow the procedure below to recover the data storage disk or the transaction log storage directory.

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance, refer to "2.1.2 Using Commands" for details.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

3. Recover the failed disk

   Replace the disk, and then recover the volume configuration information.

4. Create a storage destination directory

   - If failure occurred on the data storage disk
     Create a data storage destination directory. If a tablespace was defined, also create a directory for it.

- If failure occurred on the translation log storage disk
  Create a transaction log storage destination directory.

In [Properties] in Windows(R) Explorer, set appropriate permissions so that only the instance administrator can access the storage destination directory. (Refer to [Help and Support] in Windows(R) for information on [Properties].)

See

Refer to "Preparing Directories to Deploy Resources" under "Setup" in the Installation and Setup Guide for Server for information on how to create a storage directory.

5. Recover the keystore, and enable automatic opening of the keystore

   When the data in the database has been encrypted, restore the keystore to its state at the time of the database backup. Configure automatic opening of the keystore as necessary.

6. Recover the database cluster

   Recover the database cluster using the backup data.

   Specify the following in the pgx_rcvall command:

   - Specify the data storage location in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
   - Specify the backup data storage location in the -B option.

Example

```
> pgx_rcvall -D D:\database\inst1 -B E:\backup\inst1
```

Note

If recovery fails, remove the cause of the error in accordance with the displayed error message and then re-execute the pgx_rcvall command.

If the message "pgx_rcvall: an error occurred during recovery" is displayed, then the log recorded when recovery was executed is output after this message. The cause of the error is output in around the last fifteen lines of the log, so remove the cause of the error in accordance with the message and then re-execute the pgx_rcvall command.

The following message displayed during recovery is output as part of normal operation of pgx_rcvall command (therefore the user does not need not be concerned).

```
FATAL:  The database system is starting
```

7. Start the instance

   Refer to "2.1.2 Using Commands" for information on how to start an instance.

8. Resume applications

   Resume applications that are using the database.

If failure occurred on the backup data storage disk

The procedure for recovering the backup data storage disk is described below.

There are two methods of taking action:

- Performing recovery while the instance is active

- Stopping the instance before performing recovery

The following table shows the different steps to be performed depending on whether you stop the instance.

No	Step	Instance stopped	
		No	Yes
1	Confirm that transaction log mirroring has stopped	Y	N
2	Stop output of archive logs	Y	N
3	Stop applications	N	Y
4	Stop the instance	N	Y
5	Recover the failed disk	Y	Y
6	Create a backup data storage destination directory	Y	Y
7	Resume output of archive logs	Y	N
8	Resume transaction log mirroring	Y	N
9	Start the instance	N	Y
10	Run backup	Y	Y
11	Resume applications	N	Y

Y: Required
N: Not required

The procedure is as follows:

If an instance has not been stopped

1. Confirm that transaction log mirroring has stopped

   Use the following SQL function to confirm that transaction log mirroring has stopped.

```
 postgres=# SELECT pgx_is_wal_multiplexing_paused();
 pgx_is_wal_multiplexing_paused
-------------------
 t
(1 row)
```

   If transaction log mirroring has not stopped, then stop it using the following SQL function.

```
postgres=# SELECT pgx_pause_wal_multiplexing();
LOG:  multiplexing of transaction log files has been stopped
pgx_pause_wal_multiplexing
----------------------------

(1 row)
```

2. Stop output of archive logs

   Transaction logs may accumulate during replacement of backup storage disk, and if the data storage disk or the transaction log storage disk becomes full, there is a risk that operations may not be able to continue.

   To prevent this, use the following methods to stop output of archive logs.

   - Changing archive_command

     Specify a command that will surely complete normally, so that archive logs will be regarded as having been output.

     If you specify echo, a message is output to the server log, so it may be used as a reference when you conduct investigations.

- Reload the configuration file

  Execute the pg_ctl reload command or the pg_reload_conf SQL function to reload the configuration file.

If you simply want to stop output of errors without the risk that operations will not be able to continue, specify an empty string ('') in archive_command and reload the configuration file.

3. Recover the failed disk

   Replace the disk, and then recover the volume configuration information.

4. Create a backup data storage destination

   Create a backup data storage destination.

   In [Properties] in Windows(R) Explorer, set appropriate permissions so that only the instance administrator can access the backup data storage destination directory. (Refer to [Help and Support] in Windows(R) for information on [Properties].)

   Refer to "3.2.2 Using Server Commands" for information on how to create a backup data storage destination.

5. Resume output of archive logs

   Return the archive_command setting to its original value, and reload the configuration file.

6. Resume transaction log mirroring

   Execute the pgx_resume_wal_multiplexing SQL function.

   Example

   ```
   SELECT pgx_resume_wal_multiplexing()
   ```

7. Run backup

   Use the pgx_dmpall command to back up the database cluster.

   Specify the following value in the pgx_dmpall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

   Example

   ```
   > pgx_dmpall -D D:\database\inst1
   ```

If an instance has been stopped

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance. Refer to "2.1.2 Using Commands" for details.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

3. Recover the failed disk

   Replace the disk, and then recover the volume configuration information.

4. Create a backup data storage destination

   Create a backup data storage destination.

   In [Properties] in Windows(R) Explorer, set appropriate permissions so that only the instance administrator can access the backup data storage destination directory. (Refer to [Help and Support] in Windows(R) for information on [Properties].)

   Refer to "3.2.2 Using Server Commands" for details.

5. Start the instance

   Start the instance. Refer to "2.1.2 Using Commands" for information on how to start an instance.

6. Run backup

   Use the pgx_dmpall command to back up the database cluster.

   Specify the following value in the pgx_dmpall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

   Example

```
> pgx_dmpall -D D:\database\inst1
```

7. Resume applications

   Resume applications that are using the database.

See

- Refer to "pgx_rcvall" and "pgx_dmpall" in the Reference for information on the pgx_rcvall command and pgx_dmpall command.
- Refer to "Write Ahead Log" under "Server Administration" in the PostgreSQL Documentation for information on archive_command.
- Refer to "B.1 WAL Mirroring Control Functions" for information on pgx_resume_wal_multiplexing.

8.2 Recovering from Data Corruption

If data in a disk is logically corrupted and the database does not operate properly, you can recover the database cluster to its state at the time of backup.

There are two methods of recovery:

- 8.2.1 Using WebAdmin
- 8.2.2 Using the pgx_rcvall Command

Note

- Back up the database cluster after recovering it. Backup deletes obsolete archive logs (transaction logs copied to the backup data storage destination), freeing up disk space and reducing the recovery time.
- If you recover data to a point in the past, a new time series (database update history) will start from that recovery point. When recovery is complete, the recovery point is the latest point in the new time series. When you subsequently recover data to the latest state, the database update is re-executed on the new time series.

8.2.1 Using WebAdmin

If using WebAdmin, recover the data to the point immediately prior to data corruption by using the backup data.

Refer to "8.1.1 Using WebAdmin" for details.

8.2.2 Using the pgx_rcvall Command

Recover the database cluster by specifying in the pgx_rcvall command the date and time of the backup you want to read from. Then re-execute the transaction as required to recover the data.

Follow the procedure below to recover the data storage disk.

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance. Refer to "2.1.2 Using Commands" for information on how to stop an instance.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

3. Confirm the backup date and time

   Pinpoint a date and time prior to the data corruption based on the content of the job log or event log.

4. Recover the keystore, and enable automatic opening of the keystore

   When the data in the database has been encrypted, restore the keystore to its state at the time of the database backup. Configure automatic opening of the keystore as necessary.

5. Recover the database cluster

   Use the pgx_rcvall command to recover the database cluster.

   Specify the following values in the pgx_rcvall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
   - Specify the backup storage directory in the -B option.
   - Specify the recovery date and time in the -e option.

   Example

   In the following examples, "May 20, 2015 10:00:00" is specified as the recovery time.

```
> pgx_rcvall -D D:\database\inst1 -B E:\backup\inst1 -e "2015-05-20 10:00:00"
```

   Note

   If recovery fails, remove the cause of the error in accordance with the displayed error message and then re-execute the pgx_rcvall command.

   If the message "pgx_rcvall: an error occurred during recovery" is displayed, then the log recorded when recovery was executed is output after this message. The cause of the error is output in around the last fifteen lines of the log, so remove the cause of the error in accordance with the message and then re-execute the pgx_rcvall command.

   The following message displayed during recovery is output as part of normal operation of pgx_rcvall command (therefore the user does not need not be concerned).

```
FATAL: The database system is starting
```

6. Start the instance

   Start the instance. Refer to "2.1.2 Using Commands" for information on how to start an instance.

   If necessary, re-execute transaction processing from the specified recovery time, and then resume database operations.

   Note

   The pgx_rcvall command cannot accurately recover a hash index. If you are using a hash index, wait for the instance to start and then execute the REINDEX command for the appropriate index.

7. Resume applications

   Resume applications that are using the database.

See

Refer to "pgx_rcvall" in the Reference for information on the pgx_rcvall command.

8.3 Recovering from an Incorrect User Operation

This section describes how to recover database clusters when data has been corrupted due to erroneous user operations.

There are two methods of recovery:

- 8.3.1 Using WebAdmin
- 8.3.2 Using the pgx_rcvall Command

Note

- Back up the database cluster after recovering it. Backup deletes obsolete archive logs (transaction logs copied to the backup data storage destination), freeing up disk space and reducing the recovery time.
- If you recover data to a point in the past, a new time series (database update history) will start from that recovery point. When recovery is complete, the recovery point is the latest point in the new time series. When you subsequently recover data to the latest state, the database update is re-executed on the new time series.
- An effective restore point is one created on a time series for which you have made a backup. That is, if you recover data to a point in the past, you cannot use any restore points set after that recovery point. Therefore, once you manage to recover your target past data, make a backup.

8.3.1 Using WebAdmin

You can use WebAdmin to recover data to a backup point.

Follow the procedure below to recover the data in the data storage disk.

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance. Refer to "2.1.1 Using WebAdmin" for information on how to stop an instance.

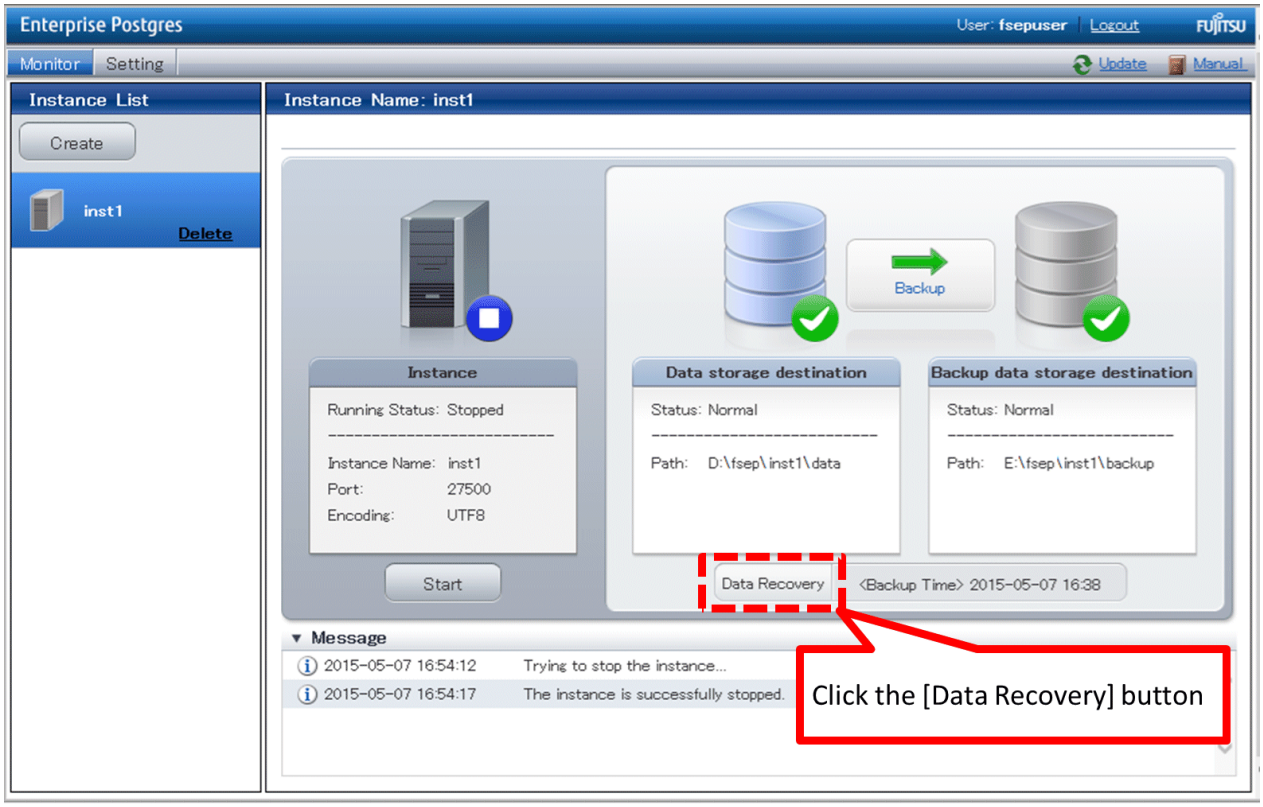
3. Recover the keystore, and enable automatic opening of the keystore

   Do the following if the data in the database has been encrypted:

   - Restore the keystore to its state at the time of the database backup.
   - Enable automatic opening of the keystore.

4. Recover the database cluster

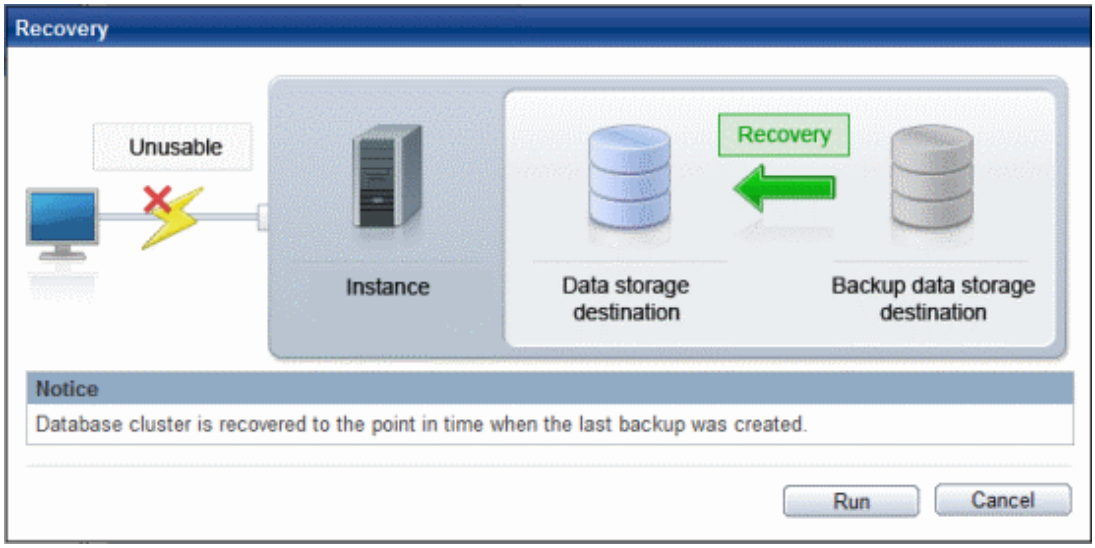
   Log in to WebAdmin, and in the [Monitor] window, click [Data Recovery].



5. Recover to the backup point

   In the [Recovery] dialog box that appears, click [Run].

   [Recovering] is displayed in the [Monitor] window, and recovery is performed. An instance is automatically started when recovery is successful.



Note

WebAdmin cannot accurately recover a hash index. If you are using a hash index, then after recovery, execute the REINDEX command for the appropriate index.

6. Resume database operations

   If necessary, re-execute transaction processing from the backup point to when an erroneous operation was performed, and then resume database operations.

8.3.2 Using the pgx_rcvall Command

The pgx_rcvall command recovers database clusters to the restore point created with the server command. Refer to "Setting a restore point" in "3.2.2 Using Server Commands" for information on how to create a restore point.

Follow the procedure below to recover the data in the data storage disk.

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance. Refer to "2.1.2 Using Commands" for information on how to stop an instance.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

3. Confirm the restore point

   Use a restore point recorded in an arbitrary file, as explained in ""3.2.2 Using Server Commands", to determine a restore point prior to the erroneous operation.

4. Recover the keystore, and enable automatic opening of the keystore

   When the data in the database has been encrypted, restore the keystore to its state at the time of the database backup. Configure automatic opening of the keystore as necessary.

5. Recover the database cluster

   Use the pgx_rcvall command to recover the database cluster.

   Specify the following values in the pgx_rcvall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
   - Specify the backup data storage destination in the -B option.
   - The -n option recovers the data to the specified restore point.

   Example

   The following example executes the pgx_rcvall command with the restore point "batch_20150503_1".

```
> pgx_rcvall -D D:\database\inst1 -B E:\backup\inst1 -n batch_20150503_1
```

   Note

   If recovery fails, remove the cause of the error in accordance with the displayed error message and then re-execute the pgx_rcvall command.

   If the message "pgx_rcvall: an error occurred during recovery" is displayed, then the log recorded when recovery was executed is output after this message. The cause of the error is output in around the last fifteen lines of the log, so remove the cause of the error in accordance with the message and then re-execute the pgx_rcvall command.

   The following message displayed during recovery is output as part of normal operation of pgx_rcvall (therefore the user does not need not be concerned).

```
FATAL:  The database system is starting
```

6. Start the instance

   Start the instance.

Refer to "2.1.2 Using Commands" for information on how to start an instance.

Note

The pgx_rcvall command cannot accurately recover a hash index. If you are using a hash index, wait for the instance to start and then execute the REINDEX command for the appropriate index.

7. Restart operation of the database

   If necessary, re-execute transaction processing from the specified recovery time to the point when an erroneous operation was performed, and then resume database operations.

See

Refer to "pgx_rcvall" in the Reference for information on the pgx_rcvall command.

8.4 Actions in Response to an Application Error

If there is a connection from a client that has been in the waiting state for an extended period, you can minimize performance degradation of the database by closing the problematic connection.

The following methods are available for identifying a connection to be closed:

- view(pg_stat_activity) (refer to "8.4.1 When using the view (pg_stat_activity)")
- pgAdmin (refer to "8.4.2 Using pgAdmin")

Use the system management function (pg_terminate_backend) to disconnect connections.

8.4.1 When using the view (pg_stat_activity)

When using the view (pg_stat_activity), follow the procedure below to close a connection.

1. Use psql command to connect to the postgres database.

```
> psql postgres
psql (9.4.4)
Type "help" for help.
```

2. Close connections from clients that have been in the waiting state for an extended period.

   Use pg_terminate_backend() to close connections that have been trying to connect for an extended period.

   However, when considering continued compatibility of applications, do not reference or use system catalogs and functions directly in SQL statements. Refer to " Notes on Application Compatibility" in the Application Development Guide for details.

   Example

   The following example closes connections where the client has been in the waiting state for at least 60 minutes.

```
select pid,usename,application_name,client_hostname,pg_terminate_backend(pid) from
pg_stat_activity where state='idle in transaction' and current_timestamp > cast(query_start +
interval '60 minutes' as timestamp);
-[ RECORD 1 ]--------+---------------
pid                  | 4684
username             | fsepuser
application_name     | apl1
client_addr          | 192.11.11.1
pg_terminate_backend | t
```

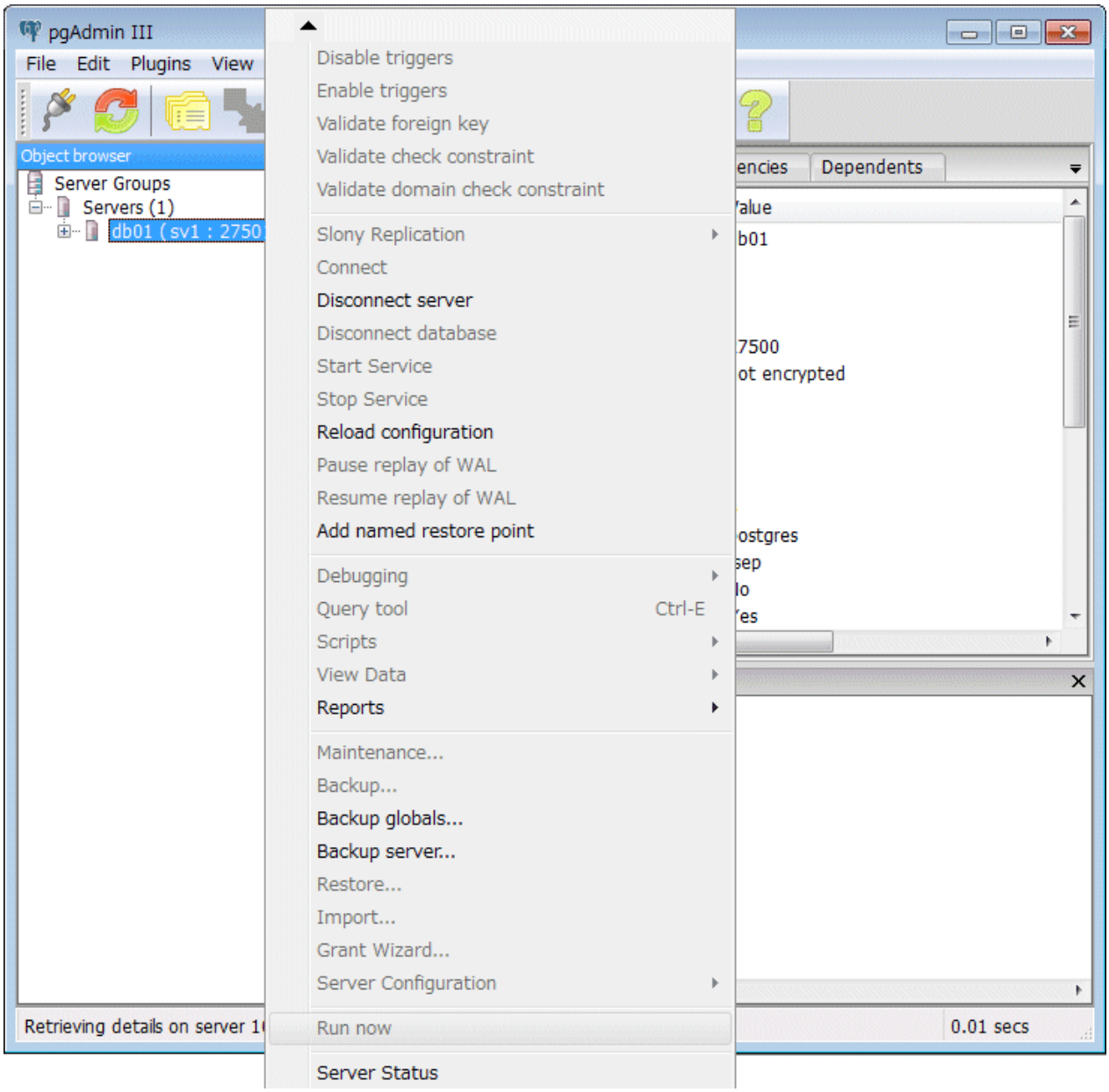
See

- Refer to "System Administration Functions" under "The SQL Language" in the PostgreSQL Documentation for information on pg_terminate_backend.
- Refer to "Notes on Application Compatibility" in the Application Development Guide for information on how to maintain application compatibility.

8.4.2 Using pgAdmin

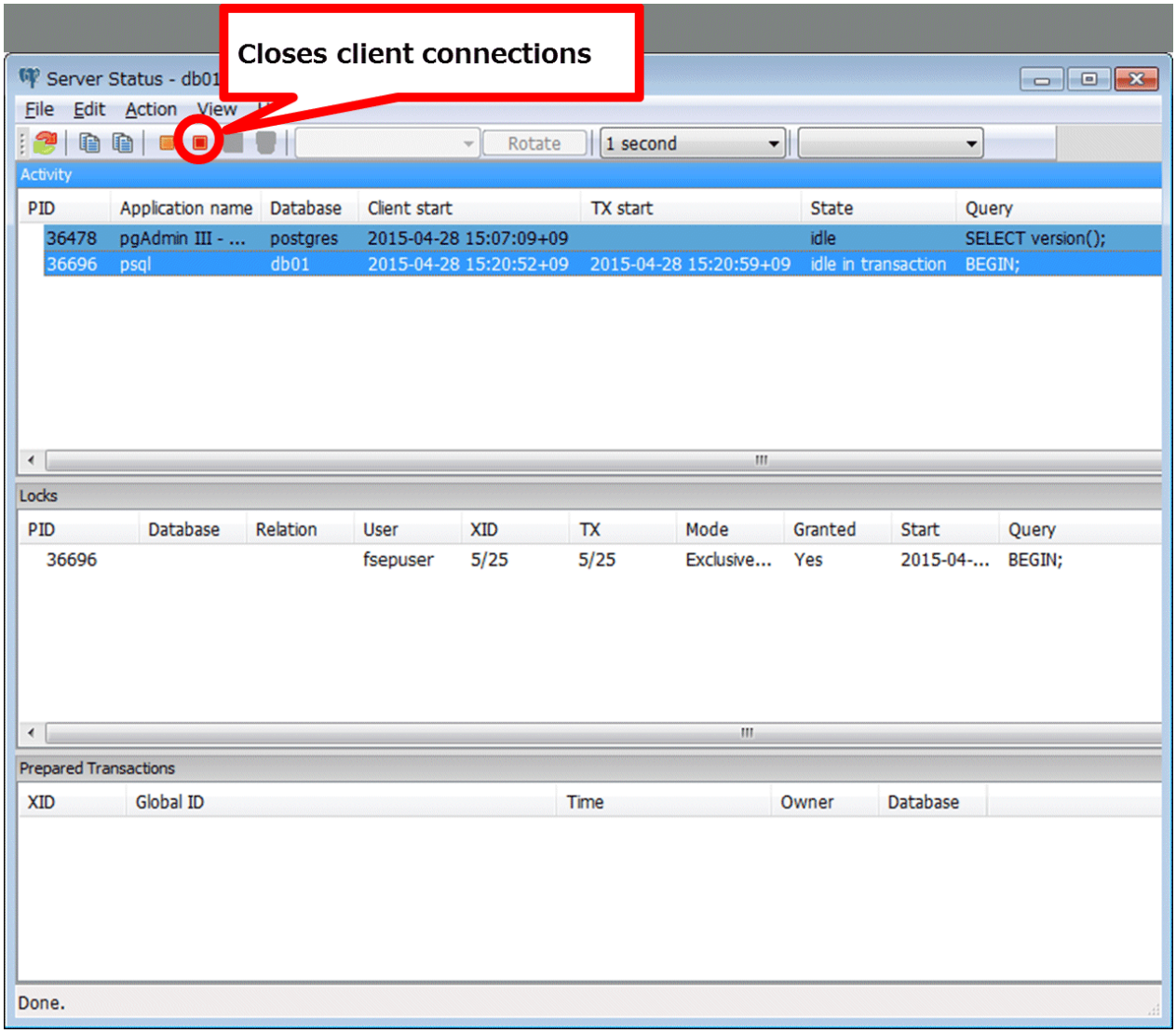
If using pgAdmin, follow the procedure below to close connections.

1. From the [Tools] menu in pgAdmin, click [Server Status].



2. Close client connections that have been in the waiting state for an extended period.

   From the transaction start time displayed under [TX Start], select connections that have been in the waiting state for an extended period. Then click the red square button to close the connections.



8.5 Actions in Response to an Access Error

If access is denied, grant privileges allowing the instance administrator to operate the following directories, and then re-execute the operation. Also, refer to the event log and the server log, and confirm that the file system has not been mounted as read-only due to a disk error. If the file system has been mounted as read-only, mount it properly and then re-execute operations.

- Data storage destination
- Tablespace storage destination
- Transaction log storage destination
- Backup data storage destination

See

Refer to "Preparing Directories to Deploy Resources" under "Setup" in the Installation and Setup Guide for Server for information on the privileges required for the directory.

8.6 Actions in Response to Insufficient Space on the Data Storage Destination

If the data storage destination runs out of space, check if the disk contains any unnecessary files and delete them so that operations can continue.

If deleting unnecessary files does not solve the problem, you must migrate data to a disk with larger capacity.

There are two methods of migrating data:

- 8.6.1 Using a Tablespace
- 8.6.2 Replacing the Disk with a Larger Capacity Disk

8.6.1 Using a Tablespace

Enterprise Postgres enables you to use a tablespace to change the storage destination of database objects, such as tables and indexes, to a different disk.

The procedure is as follows:

1. Create a tablespace

   Use the CREATE TABLESPACE command to create a new tablespace in the prepared disk.

2. Modify the tablespace

   Use the ALTER TABLE command to modify tables for the newly defined tablespace.

See

Refer to "SQL Commands" under "Reference" in the PostgreSQL Documentation for information on the CREATE TABLESPACE command and ALTER TABLE command.

8.6.2 Replacing the Disk with a Larger Capacity Disk

Before replacing the disk with a larger capacity disk, migrate resources at the data storage destination using the backup and recovery features.

There are two methods of performing backup and recovery:

- 8.6.2.1 Using WebAdmin
- 8.6.2.2 Using Server Commands

The following sections describe procedures that use each of these methods to replace the disk and migrate resources at the data storage destination.

Note

- Before replacing the disk, stop applications and instances that are using the database.
- It is recommended that you back up the database cluster following recovery. Backup deletes obsolete archive logs (transaction logs copied to the backup data storage destination), freeing up disk space and reducing the recovery time.

8.6.2.1 Using WebAdmin

Follow the procedure below to replace the disk and migrate resources at the data storage destination by using WebAdmin.

1. Back up files

   If the disk at the data storage destination contains any required files, back up the files. It is not necessary to back up the data storage destination.

2. Stop applications

   Stop applications that are using the database.

3. Back up the database cluster

   Back up the latest resources at the data storage destination. Refer to "3.2.1 Using WebAdmin" for details.

4. Stop the instance

   Stop the instance. Refer to "2.1.1 Using WebAdmin" for information on how to stop an instance.

5. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

6. Recover the database cluster

   Log in to WebAdmin, and perform recovery operations. Refer to steps 4 ("Create a tablespace directory ") to 7 ("Run recovery") under "If failure occurred in the data storage disk or the transaction log storage disk" in "8.1.1 Using WebAdmin" for information on the procedure. An instance is automatically started when recovery is successful.

7. Resume applications

   Resume applications that are using the database.

8. Restore the files

   Restore the files backed up in step 1.

8.6.2.2 Using Server Commands

Follow the procedure below to replace the disk and migrate resources at the data storage destination by using server commands.

1. Back up files

   If the disk at the data storage destination contains any required files, back up the files. It is not necessary to back up the data storage destination.

2. Stop applications

   Stop applications that are using the database.

3. Back up the database cluster

   Back up the latest resources at the data storage destination. Refer to "3.2.2 Using Server Commands" for details.

4. Stop the instance

   After backup is complete, stop the instance. Refer to "2.1.2 Using Commands" for information on how to stop an instance.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

5. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

6. Create a data storage destination

   Create a data storage destination. If a tablespace was defined, also create a directory for it.

   In [Properties] in Windows(R) Explorer, set appropriate permissions so that only the instance administrator can access the data storage destination directory. (Refer to [Help and Support] in Windows(R) for information on [Properties].)

7. Recover the keystore, and enable automatic opening of the keystore

   When the data in the database has been encrypted, restore the keystore to its state at the time of the database backup. Configure automatic opening of the keystore as necessary.

8. Recover the database cluster

   Use the pgx_rcvall command to recover the database cluster.

- Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
- Specify the backup storage directory in the -B option.

Example

```
> pgx_rcvall -D D:\database\inst1 -B E:\backup\inst1
```

Note

If recovery fails, remove the cause of the error in accordance with the displayed error message and then re-execute the pgx_rcvall command.

If the message "pgx_rcvall: an error occurred during recovery" is displayed, then the log recorded when recovery was executed is output after this message. The cause of the error is output in around the last fifteen lines of the log, so remove the cause of the error in accordance with the message and then re-execute the pgx_rcvall command.

The following message displayed during recovery is output as part of normal operation of pgx_rcvall (therefore the user does not need not be concerned).

```
FATAL:  The database system is starting
```

See

Refer to "pgx_rcvall" in the Reference for information on the pgx_rcvall command.

9. Start the instance

   Start the instance.

   Refer to "2.1.2 Using Commands" for information on how to start an instance.

   ### Note

   The pgx_rcvall command cannot accurately recover a hash index. If you are using a hash index, wait for the pgx_rcvall command to end and then execute the REINDEX command for the appropriate index.

10. Resume applications

    Resume applications that are using the database.

11. Restore files

    Restore the files backed up in step 1.

8.7 Actions in Response to Insufficient Space on the Backup Data Storage Destination

If space runs out on the backup data storage destination, check if the disk contains any unnecessary files and delete them, and then make a backup as required.

If deleting unnecessary files does not solve the problem, take the following action:

- 8.7.1 Temporarily Saving Backup Data
- 8.7.2 Replacing the Disk with a Larger Capacity Disk

8.7.1 Temporarily Saving Backup Data

This method involves temporarily moving backup data to a different directory, saving it there, and securing disk space on the backup data storage destination so that a backup can be made normally.

Use this method if you need time to prepare a larger capacity disk.

If space runs out on the backup data storage destination, archive logs can no longer be stored in the backup data storage destination. As a result, transaction logs continue to accumulate in the data storage destination or the transaction log storage destination.

If action is not taken soon, the transaction log storage destination will become full, and operations may not be able to continue.

To prevent this, secure space in the backup data storage destination, so that archive logs can be stored.

There are two methods of taking action:

- 8.7.1.1 Using WebAdmin
- 8.7.1.2 Using Server Commands

8.7.1.1 Using WebAdmin

Follow the procedure below to recover the backup data storage disk.

1. Temporarily save backup data

   Move backup data to a different directory and temporarily save it, and secure space in the backup data storage destination directory.

   The reason for saving the backup data is so that the data in the data storage destination can be recovered even if it is corrupted before you perform recovery. If there is no disk at the save destination and you consider that there is no risk of corruption at the data storage destination, delete the backup data.

   The following example saves backup data from the backup data storage destination directory (E:\backup\inst1) under F:\mnt\usb\backup.

   Example

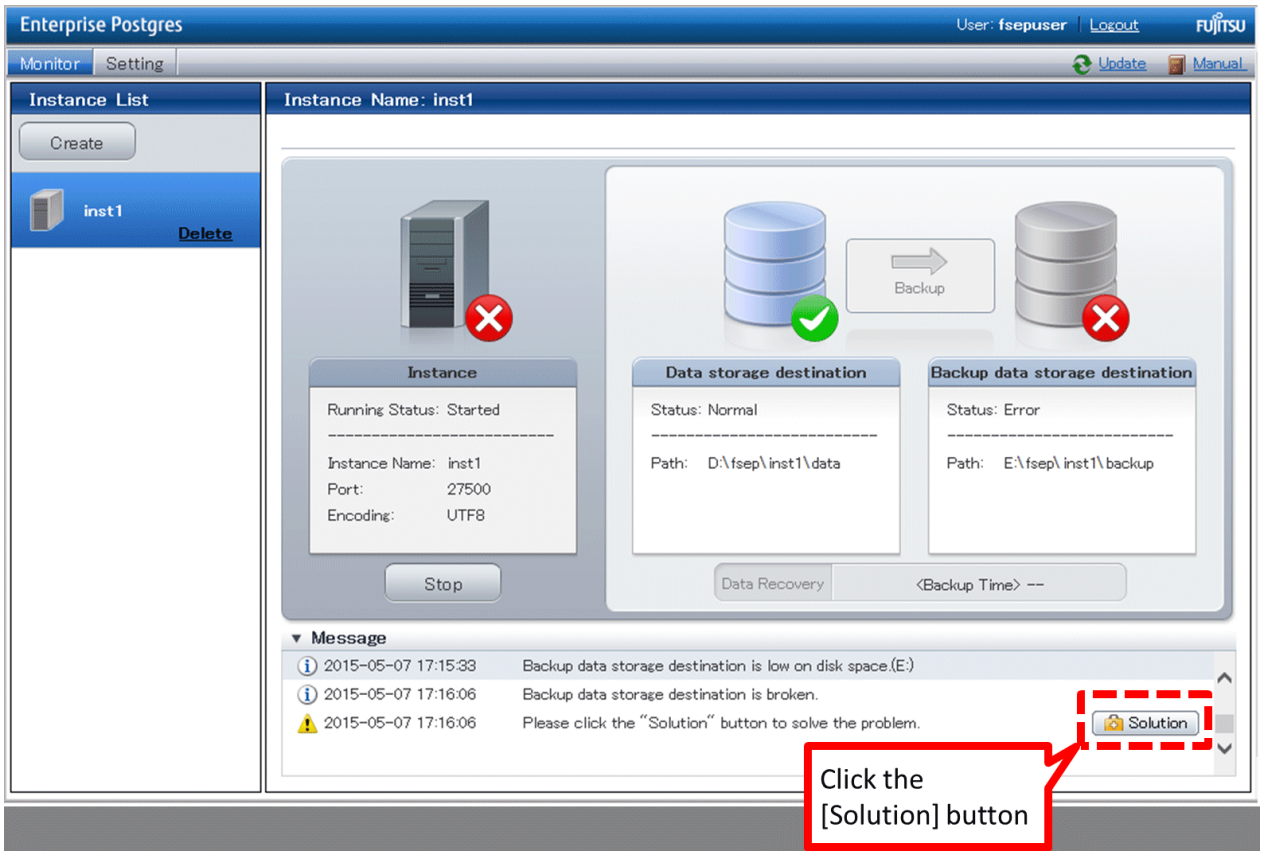
   ```
   > mkdir F:\mnt\usb\backup
   > move E:\backup\inst1\* F:\mnt\usb\backup
   ```

   Note: Place the temporary backup destination directory in a location where it will not impact on operating system resources or Enterprise Postgres resources.

2. Recover backup data

   Log in to WebAdmin and start recovering backup data.

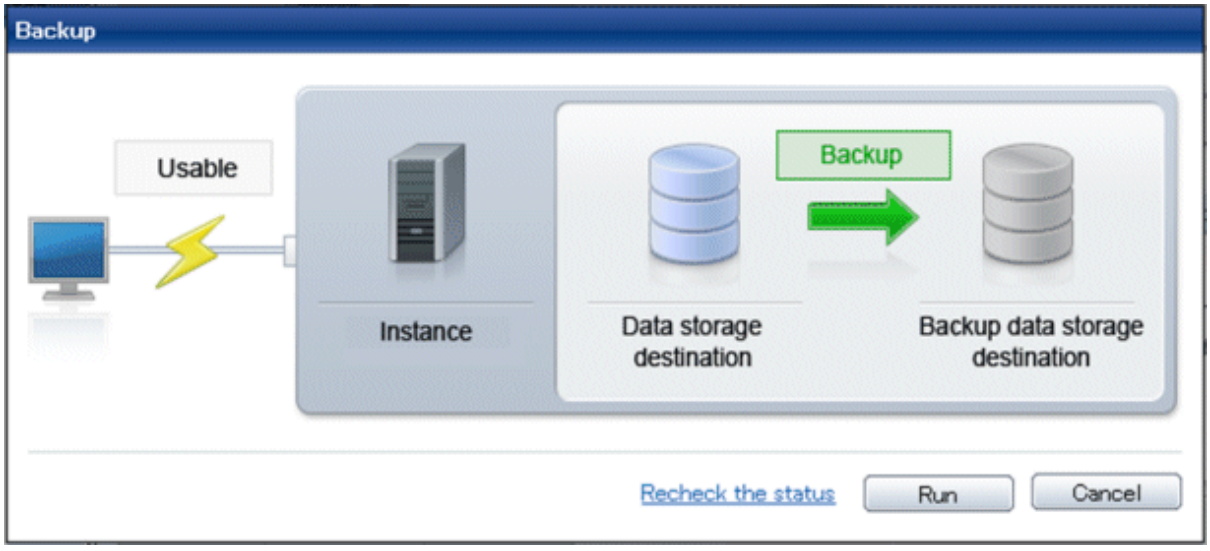
In the [Monitor] window, click [Solution] for the error message.



The above screen is displayed if moving and saving backup data causes WebAdmin to detect that the content at the backup data storage destination is missing. If you use another method to save the data and no abnormality is detected, click [->] next to the "Backup" caption in the [Monitor] menu window.

3. Run backup

   Perform the backup to enable the recovery of backup data. In the [Backup] dialog box displayed, click [Run]. [Backuping] is displayed in the [Monitor] window and the backup is performed. An instance is automatically activated when backup is performed.



4. Delete temporarily saved backup data

   If backup completes normally, the temporarily saved backup data becomes unnecessary and is deleted.

The following example deletes backup data that was temporarily saved in F:\mnt\usb.

Example

```
> rmdir /S /Q F:\mnt\usb\backup
```

8.7.1.2 Using Server Commands

The following describes the procedure for recovering the backup storage disk.

There are two methods of taking action:

- Performing recovery while the instance is active
- Stopping the instance before performing recovery

The following table shows the different steps to be performed depending on whether you stop the instance.

No	Step	Instance stopped	
		No	Yes
1	Stop transaction log mirroring	Y	N
2	Stop output of archive logs	Y	N
3	Stop applications	N	Y
4	Stop the instance	N	Y
5	Temporarily save backup data	Y	Y
6	Resume output of archive logs	Y	N
7	Resume transaction log mirroring	Y	N
8	Start an instance	N	Y
9	Run backup	Y	Y
10	Resume applications	N	Y
11	Delete temporarily saved backup data	Y	Y

Y: Required
N: Not required

The procedure is as follows:

Performing recovery while the instance is active

1. Stop transaction log mirroring

   Stop transaction log mirroring.

   ```
   postgres=# SELECT pgx_pause_wal_multiplexing();
   LOG:  multiplexing of transaction log files has been stopped
   pgx_pause_wal_multiplexing
   ---------------------------

   (1 row)
   ```

2. Stop output of archive logs

   Transaction logs may accumulate during replacement of backup storage disk, and if the data storage disk or the transaction log storage disk becomes full, there is a risk that operations may not be able to continue.

   To prevent this, use the following methods to stop output of archive logs.

- Changing the archive_command parameter

  Specify a command that will surely complete normally, so that archive logs will be regarded as having been output.

  If you specify echo, a message is output to the server log, so it may be used as a reference when you conduct investigations.

- Reloading the configuration file

  Run the pg_ctl reload command or the pg_reload_conf SQL function.

If you simply want to stop output of errors without the risk that operations will not be able to continue, specify an empty string ('') in archive_command and reload the configuration file.

3. Temporarily save backup data

   Move backup data to a different directory and temporarily save it, and secure space in the backup data storage destination directory.

   The reason for saving the backup data is so that the data in the data storage destination can be recovered even if it is corrupted before you perform the next step. If there is no disk at the save destination and you consider that there is no risk of corruption at the data storage destination, delete the backup data.

   The following example saves backup data from the backup data storage destination directory (E:\backup\inst1) under F:\mnt\usb \backup.

   Example

   ```
   > mkdir F:\mnt\usb\backup
   > move E:\backup\inst1\* F:\mnt\usb\backup
   ```

   Note: Place the temporary backup destination directory in a location where it will not impact on operating system resources or Enterprise Postgres resources.

4. Resume output of archive logs

   Return the archive_command setting to its original value, and reload the configuration file.

5. Resume transaction log mirroring

   Execute the pgx_resume_wal_multiplexing SQL function.

   Example

   ```
   SELECT pgx_resume_wal_multiplexing()
   ```

6. Run backup

   Use the pgx_dmpall command to back up the database cluster.

   Specify the following option in the pgx_dmpall command:

   - Specify the directory of the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

     Example

     ```
     > pgx_dmpall -D D:\database\inst1
     ```

7. Delete temporarily saved backup data

   If backup completes normally, the temporarily saved backup data becomes unnecessary and is deleted.

   The following example deletes backup data that was temporarily saved in F:\mnt\usb.

   Example

   ```
   > rmdir /S /Q F:\mnt\usb\backup
   ```

If an instance has been stopped

1. Stop applications

   Stop applications that are using the database.

2. Stop the instance

   Stop the instance. Refer to "2.1.2 Using Commands" for details.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

3. Temporarily save backup data

   Move backup data to a different directory and temporarily save it, and secure space in the backup data storage destination directory.

   The reason for saving the backup data is so that the data in the data storage destination can be recovered even if it is corrupted before you perform recovery. If there is no disk at the save destination and you consider that there is no risk of corruption at the data storage destination, delete the backup data.

   The following example saves backup data from the backup data storage destination directory (E:\backup\inst1) under F:\mnt\usb\backup.

   Example

   ```
   > mkdir F:\mnt\usb\backup
   > move E:\backup\inst1\* F:\mnt\usb\backup
   ```

   Note: Place the temporary backup destination directory in a location where it will not impact on operating system resources or Enterprise Postgres resources.

4. Start the instance

   Start the instance. Refer to "2.1.2 Using Commands" for information on how to start an instance.

5. Run backup

   Use the pgx_dmpall command to back up the database cluster.

   Specify the following value in the pgx_dmpall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

     Example

     ```
     > pgx_dmpall -D D:\database\inst1
     ```

6. Resume applications

   Resume applications that are using the database.

7. Delete temporarily saved backup data

   If backup completes normally, the temporarily saved backup data becomes unnecessary and is deleted.

   The following example deletes backup data that was temporarily saved in F:\mnt\usb.

   Example

   ```
   > rmdir /S /Q F:\mnt\usb\backup
   ```

See

- Refer to "pgx_rcvall" and "pgx_dmpall" in the Reference for information on the pgx_rcvall command and pgx_dmpall command.
- Refer to "Write Ahead Log" under "Server Administration" in the PostgreSQL Documentation for information on archive_command.

- Refer to "B.1 WAL Mirroring Control Functions" for information on the pgx_is_wal_multiplexing_paused and pgx_resume_wal_multiplexing.

8.7.2 Replacing the Disk with a Larger Capacity Disk

This method involves replacing the disk at the backup data storage destination with a larger capacity disk, so that it does not run out of free space again. After replacing the disk, back up data to obtain a proper backup.

There are two methods of performing backup:

- 8.7.2.1 Using WebAdmin
- 8.7.2.2 Using Server Commands

Before replacing the disk, stop applications that are using the database.

8.7.2.1 Using WebAdmin

Follow the procedure below to recover the backup storage disk.

1. Back up files

   If the disk at the backup data storage destination contains any required files, back up the files. It is not necessary to back up the backup data storage destination.

2. Temporarily save backup data

   Save the backup data to a different directory.

   The reason for saving the backup data is so that the data in the data storage destination can be recovered even if it is corrupted before you perform the next step. If there is no disk at the save destination and you consider that there is no risk of corruption at the data storage destination, delete the backup data.

   The following example saves backup data from the backup data storage destination directory (E:\backup\inst1) under F:\mnt\usb \backup.

   Example

   ```
   > mkdir F:\mnt\usb\backup
   > move E:\backup\inst1\* F:\mnt\usb\backup
   ```

   Note: Place the temporary backup destination directory in a location where it will not impact on operating system resources or Enterprise Postgres resources.

3. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

4. Run backup

   Log in to WebAdmin, and perform recovery operations. Refer to steps 2 ("Recover the backup data") and 3 ("Run backup") under "If failure occurred on the backup storage disk" in "8.1.1 Using WebAdmin".

5. Restore files

   Restore the files backed up in step 1.

6. Delete temporarily saved backup data

   If backup completes normally, the temporarily saved backup data becomes unnecessary and is deleted.

   The following example deletes backup data that was temporarily saved in F:\mnt\usb.

Example

```
> rmdir /S /Q F:\mnt\usb\backup
```

8.7.2.2 Using Server Commands

The procedure for recovering the backup data storage disk is described below.

There are two methods of taking action:

- Performing recovery while the instance is active
- Stopping the instance before performing recovery

The following table shows the different steps to be performed depending on whether you stop the instance.

No	Step	Instance stopped	
		No	Yes
1	Back up files	Y	Y
2	Temporarily save backup data	Y	Y
3	Confirm that transaction log mirroring has stopped	Y	N
4	Stop output of archive logs	Y	N
5	Stop applications	N	Y
6	Stop the instance	N	Y
7	Replace with a larger capacity disk	Y	Y
8	Create a backup storage directory	Y	Y
9	Resume output of archive logs	Y	N
10	Resume transaction log mirroring	Y	N
11	Start the instance	N	Y
12	Run backup	Y	Y
13	Resume applications	N	Y
14	Restore files	Y	Y
15	Delete temporarily saved backup data	Y	Y

Y: Required
N: Not required

The procedure is as follows:

If an instance has not been stopped

1. Back up files

   If the disk at the backup data storage destination contains any required files, back up the files. It is not necessary to back up the backup data storage destination.

2. Temporarily save backup data

   Save the backup data to a different directory.

   The reason for saving the backup data is so that the data in the data storage destination can be recovered even if it is corrupted before you perform the next step. If there is no disk at the save destination and you consider that there is no risk of corruption at the data storage destination, delete the backup data.

   The following example saves backup data from the backup data storage destination directory (E:\backup\inst1) under F:\mnt \usb\backup.

Example

```
> mkdir F:\mnt\usb\backup
> move E:\backup\inst1\* F:\mnt\usb\backup
```

3. Confirm that transaction log mirroring has stopped

   Use the following SQL function to confirm that transaction log mirroring has stopped.

```
 postgres=# SELECT pgx_is_wal_multiplexing_paused();
 pgx_is_wal_multiplexing_paused
-------------------
 t
(1 row)
```

   If transaction log mirroring has not stopped, then stop it using the following SQL function.

```
postgres=# SELECT pgx_pause_wal_multiplexing();
LOG:  multiplexing of transaction log files has been stopped
pgx_pause_wal_multiplexing
---------------------------

(1 row)
```

4. Stop output of archive logs

   Transaction logs may accumulate during replacement of backup storage disk, and if the data storage destination disk or the transaction log storage destination disk becomes full, there is a risk that operations may not be able to continue.

   To prevent this, use the following methods to stop output of archive logs.

   - Changing the archive_command parameter

     Specify a command that will surely complete normally, so that archive logs will be regarded as having been output.

     If you specify echo, a message is output to the server log, so it may be used as a reference when you conduct investigations.

   - Reloading the configuration file

     Run the pg_ctl reload command or the pg_reload_conf SQL function.

   If you simply want to stop output of errors without the risk that operations will not be able to continue, specify an empty string ('') in archive_command and reload the configuration file.

5. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

6. Create a backup data storage destination

   Create a backup data storage destination.

   In [Properties] in Windows(R) Explorer, set appropriate permissions so that only the instance administrator can access the backup data storage destination directory. (Refer to [Help and Support] in Windows(R) for information on [Properties].)

   Refer to "3.2.2 Using Server Commands" for details.

7. Resume output of archive logs

   Return the archive_command setting to its original value, and reload the configuration file.

8. Resume transaction log mirroring

   Execute the pgx_resume_wal_multiplexing SQL function.

   Example

```
SELECT pgx_resume_wal_multiplexing()
```

9. Run backup

   Use the pgx_dmpall command to back up the database cluster.

   Specify the following value in the pgx_dmpall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

   Example

```
> pgx_dmpall -D D:\database\inst1
```

10. Restore files

    Restore the files backed up in step 1.

11. Delete temporarily saved backup data

    If backup completes normally, the temporarily saved backup data becomes unnecessary and is deleted.

    The following example deletes backup data that was temporarily saved in F:\mnt\usb.

    Example

```
> rmdir /S /Q F:\mnt\usb\backup
```

If an instance has been stopped

1. Back up files

   If the disk at the backup data storage destination contains any required files, back up the files. It is not necessary to back up the backup data storage destination.

2. Temporarily save backup data

   Save the backup data to a different directory.

   The reason for saving the backup data is so that the data in the data storage destination can be recovered even if it is corrupted before you perform the next step. If there is no disk at the save destination and you consider that there is no risk of corruption at the data storage destination, delete the backup data.

   The following example saves backup data from the backup data storage destination directory (E:\backup\inst1) under F:\mnt\usb\backup.

   Example

```
> mkdir F:\mnt\usb\backup
> move E:\backup\inst1\* F:\mnt\usb\backup
```

   Note: Place the temporary backup destination directory in a location where it will not impact on operating system resources or Enterprise Postgres resources.

3. Stop applications

   Stop applications that are using the database.

4. Stop the instance

   Stop the instance. Refer to "2.1.2 Using Commands" for information on how to stop an instance.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

5. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

6. Create a backup data storage destination

   Create a backup data storage destination.

In [Properties] in Windows(R) Explorer, set appropriate permissions so that only the instance administrator can access the backup data storage destination directory. (Refer to [Help and Support] in Windows(R) for information on [Properties].)

Refer to "3.2.2 Using Server Commands" for details.

7. Start the instance

   Start the instance. Refer to "2.1.2 Using Commands" for information on how to start an instance.

8. Run backup

   Use the pgx_dmpall command to back up the database cluster.

   Specify the following value in the pgx_dmpall command:

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.

   Example

   ```
   > pgx_dmpall -D D:\database\inst1
   ```

9. Resume applications

   Resume applications that are using the database.

10. Restore files

    Restore the files backed up in step 1.

11. Delete temporarily saved backup data

    If backup completes normally, the temporarily saved backup data becomes unnecessary and is deleted.

    The following example deletes backup data that was temporarily saved in F:\mnt\usb.

    Example

    ```
    > rmdir /S /Q F:\mnt\usb\backup
    ```

See

- Refer to "pgx_rcvall" and "pgx_dmpall" in the Reference for information on the pgx_rcvall command and pgx_dmpall command.
- Refer to "Write Ahead Log" under "Server Administration" in the PostgreSQL Documentation for information on archive_command.
- Refer to "B.1 WAL Mirroring Control Functions" for information on the pgx_is_wal_multiplexing_paused and pgx_resume_wal_multiplexing.

8.8 Actions in Response to Insufficient Space on the Transaction Log Storage Destination

If the transaction log storage destination runs out of space, check if the disk contains any unnecessary files and delete them so that operations can continue.

If deleting unnecessary files does not solve the problem, you must migrate data to a disk with larger capacity.

8.8.1 Replacing the Disk with a Larger Capacity Disk

Before replacing the disk with a larger capacity disk, migrate resources at the transaction log storage destination using the backup and recovery features.

There are two methods of performing backup and recovery:

- 8.8.1.1 Using WebAdmin
- 8.8.1.2 Using Server Commands

The following sections describe procedures that use each of these methods to replace the disk and migrate resources at the transaction log storage destination.

- Before replacing the disk, stop applications that are using the database.
- It is recommended that you back up the database cluster following recovery. Backup deletes obsolete archive logs (transaction logs copied to the backup data storage destination), freeing up disk space and reducing the recovery time.

8.8.1.1 Using WebAdmin

Follow the procedure below to replace the disk and migrate resources at the transaction log storage destination by using WebAdmin.

1. Back up files

   If the disk at the transaction log storage destination contains any required files, back up the files. It is not necessary to back up the transaction log storage destination.

2. Back up the database cluster

   Back up the latest data storage destination resources and transaction log storage destination resources (refer to "3.2.1 Using WebAdmin" for details).

3. Stop applications

   Stop applications that are using the database.

4. Stop the instance

   Stop the instance. Refer to "2.1.1 Using WebAdmin" for information on how to stop an instance. WebAdmin automatically stops instances if recovery of the database cluster is performed without stopping the instance.

5. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

6. Create a tablespace directory

   If a tablespace was defined after backing up, create a directory for it.

7. Recover the keystore, and enable automatic opening of the keystore

   Do the following if the data in the database has been encrypted:

   - Restore the keystore to its state at the time of the database backup.
   - Enable automatic opening of the keystore.

8. Recover the database cluster

   Log in to WebAdmin, and perform recovery operations. Refer to steps 4 ("Create a tablespace directory ") to 7 ("Run Recovery") under " If failure occurred in the data storage disk or the transaction log storage disk " in "8.1.1 Using WebAdmin" for information on the procedure. An instance is automatically started when recovery is successful.

9. Resume applications

   Resume applications that are using the database.

10. Restore files

    Restore the files backed up in step 1.

8.8.1.2 Using Server Commands

Follow the procedure below to replace the disk and migrate resources at the transaction log storage destination by using server commands.

1. Back up files

   If the disk at the transaction log storage destination contains any required files, back up the files. It is not necessary to back up the transaction log storage destination.

2. Back up the database cluster

   Use server commands to back up the latest data storage destination resources and transaction log storage destination resources. Refer to "3.2.2 Using Server Commands" for information on how to perform backup.

3. Stop applications

   Stop applications that are using the database.

4. Stop the instance

   After backup is complete, stop the instance. Refer to "2.1.2 Using Commands" for information on how to stop an instance.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

5. Replace with a larger capacity disk

   Replace the disk. Then, recover the volume configuration information.

6. Create a transaction log storage destination

   Create a transaction log storage destination. If a tablespace was defined, also create a directory for it.

   In [Properties] in Windows(R) Explorer, set appropriate permissions so that only the instance administrator can access the transaction log destination directory. (Refer to [Help and Support] in Windows(R) for information on [Properties].)

7. Recover the keystore, and enable automatic opening of the keystore

   When the data in the database has been encrypted, restore the keystore to its state at the time of the database backup. Configure automatic opening of the keystore as necessary.

8. Recover the database cluster

   Use the pgx_rcvall command to recover the database cluster.

   - Specify the data storage destination in the -D option. If the -D option is omitted, the value of the PGDATA environment variable is used by default.
   - Specify the backup storage directory in the -B option.

   Example

```
> pgx_rcvall -D D:\database\inst1 -B E:\backup\inst1
```

Note

If recovery fails, remove the cause of the error in accordance with the displayed error message and then re-execute the pgx_rcvall command.

If the message "pgx_rcvall: an error occurred during recovery" is displayed, then the log recorded when recovery was executed is output after this message. The cause of the error is output in around the last fifteen lines of the log, so remove the cause of the error in accordance with the message and then re-execute the pgx_rcvall command.

The following message displayed during recovery is output as part of normal operation of pgx_rcvall command (therefore the user does not need not be concerned).

```
FATAL: The database system is starting
```

See

Refer to "pgx_rcvall" in the Reference for information on the pgx_rcvall command.

9. Start the instance

   Start the instance.

   Refer to "2.1.2 Using Commands" for information on how to start an instance.

   Note

   The pgx_rcvall command cannot accurately recover a hash index. If you are using a hash index, wait for the instance to start and then execute the REINDEX command for the appropriate index.

10. Resume applications

    Resume applications that are using the database.

11. Restore files

    Restore the files backed up in step 1.

8.9 Errors in More Than One Storage Disk

If an error occurs in the storage destination disks or resources are corrupted, determine the cause of the error from event logs and server logs and remove the cause.

If errors occur in either of the following combinations, you cannot recover the database.

Recreate the instance, and rebuild the runtime environment.

Data storage disk	Transaction log storage disk	Backup data storage disk
Error	-	Error
-	Error	Error

See

Refer to "Setup" in the Installation and Setup Guide for Server for information on how to create an instance and build the runtime environment.

8.10 Actions in Response to Instance Startup Failure

If an instance fails to start, refer to the event log and the server log, and determine the cause of the failure.

If using WebAdmin, remove the cause of the error. Then, click [Solution] and [Recheck the status] and confirm that the instance is in the normal state.

The following sections describe common causes of errors and the actions to take.

8.10.1 Errors in the Configuration File

If you have directly edited the configuration file using a text editor or changed the settings using WebAdmin, refer to the event log and the server log, confirm that no messages relating to the files below have been output.

- postgresql.conf
- pg_hba.conf

See

Refer to the following for information on the parameters in the configuration file:

- "Configuring Parameters" in the Installation and Setup Guide for Server
- "Appendix A Parameters"
- "Server Configuration" and "Client Authentication" under "Server Administration" in the PostgreSQL Documentation

8.10.2 Errors Caused by Power Failure or Mounting Issues

If mounting is cancelled after restarting the server, for example, because the disk device for each storage destination disk was not turned on, or because automatic mounting has not been set, then starting an instance will fail.

Refer to "8.13.2 Errors Caused by Power Failure or Mounting Issues", and take actions accordingly.

8.10.3 Errors Caused by Failure to Start PL/extJava

If an error occurs during instance startup because PL/extJava has failed to start, check the error messages output to the logs below and investigate the cause of the error by referring to "Messages Output by PL/extJava" in Messages. Refer to "Appendix G PL/extJava Log Information" for information on the location of these logs.

- Event log
- Enterprise Postgres server log
- Container server log
- Container Java VM log
- Domain server log
- Domain Java VM log

Some common causes of this error (and their corresponding corrective actions) are listed below:

- A fault may have occurred on the disk where the domain root is placed, or the disk may have becomes corrupted by an incorrect operation.
  If that is the case, perform the following procedure to restore it

  If a backup of PL/extJava has been obtained

  Refer to "7.3.7.2 Restore Method" and restore PL/extJava accordingly.

  If a backup of PL/extJava has not been obtained

  Perform the following procedure.

  1. Stop applications

     Stop applications that are using the database.

  2. Stop the instance

     Stop the instance, refer to 2.1 Starting and Stopping an Instance" for details.

     If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

  3. Recover the failed disk

     If the disk is defective, replace the disk, and then recover the volume configuration information.

  4. Delete the domain root directory

  5. Change the settings in the postgresql.conf file

     Delete the following parameters from the postgresql.conf file.

     - plextjava.http_port
     - plextjava.start_command
     - plextjava.stop_command
     - plextjava.forcible_stop_command

6. Delete the domain root information

   If "FUJITSU PCMI(isje6-*componentId*)" remains in services, execute the sc command as shown below to delete the "FUJITSU PCMI(isje6-*componentId*)" service.

   ```
   > sc delete "FUJITSU PCMI(isje6-componentId)"
   ```

   Delete the files listed below.

   - *Enterprise PostgresInstallDir*/java/etc/domain.conf
   - *Enterprise PostgresInstallDir*/java/etc/domain1_container.conf

   Example

   componentId has the following format:

   ```
   STFFSEPDB + versionLevel + firstDigitOfArchitecture
   ```

   - *versionLevel*: Version/level of Enterprise Postgres
   - *firstDigitOfArchitecture: First digit of product architecture, that is "3" for the 32-bit version, and "6" for the 64-bit version*

   For Enterprise Postgres version "0941" with 64-bit product architecture, componentId will be:.

   ```
   STFFSEPDB09416
   ```

7. Recreate PL/extJava

   Recreate PL/extJava after executing the pgx_jadmin command for PL/extJava and storing the Java applications. Refer to "Chapter 7 Setting up and Operating PL/extJava " for information on creating PL/extJava.

8. Start the instance

   Refer to 2.1 Starting and Stopping an Instance" for information on how to start an instance.

   If the instance fails to stop, refer to "8.10 Actions in Response to Instance Startup Failure".

9. Resume applications

   Resume applications that are using the database.

- If a message was output indicating that the address cannot be assigned to the container server log or the domain server log, the specified port number may be in use by another server.
  If that is the case, change the port numbers and then restart the instance (refer to "7.2.1 Preparing Port Numbers" and "7.3.3.3 Changing Port Numbers" for details).

- If a message was output to the server log of Enterprise Postgres indicating that the Java EE DAS service has failed to start, but a corresponding message indicating the cause of the failure has not been output to the domain server log, the cached information may not be up to date.
  If that is the case, delete the file below and then restart the instance.

  ```
  domainRoot\domains\domain1\osgi-cache\felix
  ```

- If a message was output indicating that there are PL/extJava processes already running, there may be processes still running after the PCMI service that monitors processes ended in an error.
  If that is the case, perform the following procedure to forcibly terminate the unnecessary processes, and then restart the instance.

  1. End server instances(Java VM)

     Forcibly terminate any processes listed in the file below and that are still running, then delete the file (perform this step for all server instances(Java VM)).

     ```
     domainRoot\nodes\localhost-domain1\serverInstance(JavaVM)Name\config\pid
     ```

2. End Java EE DAS service processes

   Forcibly terminate any processes listed in the file below and that are still running, then delete the file.

   ```
   domainRoot\domains\domain1\config\pid
   ```

3. End web server processes

   Forcibly terminate any processes listed in the file below and that are still running, then delete the file.

   ```
   Enterprise PostgresInstallDir\java\ahs22\plextjava\domain1\logs\httpd.pid
   ```

4. End PCMI service processes

   Use the method below to check if there are PCMI service processes still running - if there are, forcibly terminate them, and then restart the instance.

   - Windows Task Manager

8.10.4 Other Errors

This section describes the recovery procedure to be used if you cannot take any action or the instance cannot start even after you have referred to the event log and the server log.

There are two methods of recovery:

- 8.10.4.1 Using WebAdmin
- 8.10.4.2 Using Server Commands

Note that recovery will not be possible if there is an error at the backup data storage destination. If the problem cannot be resolved, contact Fujitsu technical support.

8.10.4.1 Using WebAdmin

Follow the procedure below to perform recovery.

1. Delete the data storage destination directory and the transaction log storage destination directory

   Back up the data storage destination directory and the transaction log storage destination directory before deleting them.

2. Reconfirm the status

   Log in to WebAdmin, and in the [Monitor] window, click the [Solution] button for the error message.

   Click [Recheck the status] to reconfirm the storage destination resources.

3. Run recovery

   Restore the database cluster after WebAdmin detects an error.

   Refer to "8.2.1 Using WebAdmin" for details.

8.10.4.2 Using Server Commands

Follow the procedure below to recover the database.

1. Delete the data storage destination directory and the transaction log storage destination directory

   Save the data storage destination directory and the transaction log storage destination directory, and then delete them.

2. Execute recovery

   Use the pgx_rcvall command to recover the database cluster.

   Refer to "8.2.2 Using the pgx_rcvall Command" for details.

8.11 Actions in Response to Failure to Stop an Instance

If an instance fails to stop, refer to the event log and the server log, and determine the cause of the failure.

If the instance cannot stop despite taking action, perform the following operation to stop the instance.

There are two methods of recovery:

- 8.11.1 Using WebAdmin
- 8.11.2 Using Server Commands

8.11.1 Using WebAdmin

Click [Stop] in the [Monitor] window and select the Fast stop mode or the Immediate stop mode to stop the instance. Forcibly terminate the server process from WebAdmin if the instance cannot be stopped.

Refer to "2.1.1 Using WebAdmin" for information on the stop modes.

8.11.2 Using Server Commands

There are three methods:

- Stopping the Instance Using the Fast Mode

  If backup is in progress, then terminate it, roll back all executing transactions, forcibly close client connections, and then stop the instance.

- Stopping the Instance Using the Immediate Mode

  Forcibly terminate the instance immediately. A crash recovery is run when the instance is restarted.

- Forcibly Stopping the Server Process

  Reliably stops the server process when the other methods are unsuccessful.

8.11.2.1 Stopping the Instance Using the Fast Mode

Specify "-m fast" in the pg_ctl command to stop the instance.

If the instance fails to stop when you use this method, stop the instance as described in "8.11.2.2 Stopping the Instance Using the Immediate Mode" or "8.11.2.3 Forcibly Stopping the Server Process".

Example

```
> pg_ctl stop -D D:\database\inst1 -m fast
```

8.11.2.2 Stopping the Instance Using the Immediate Mode

Specify "-m immediate " in the pg_ctl command to stop the instance.

If the instance fails to stop when you use this method, stop the instance as described in "8.11.2.3 Forcibly Stopping the Server Process".

Example

```
> pg_ctl stop -D D:\database\inst1 -m immediate
```

8.11.2.3 Forcibly Stopping the Server Process

If both the Fast mode and the Immediate mode fail to stop the instance, use the kill parameter of the pg_ctl command to forcibly stop the server process.

The procedure is as follows:

1. Execute the wmic command to identify the process ID of the server process.

```
c:\>wmic
wmic:root\cli>process where "name = \"postgres.exe\"" get CommandLine,Name,ProcessId
CommandLine                                                                    Name
ProcessId
:
"C:\Program Files\Fujitsu\fsepserver64\bin\postgres.exe" -D "D:\database\inst1"
postgres.exe  896
:
```

   The postgres.exe process ID(896) that indicates the data storage destination directory of the applicable instance in the -D option becomes the server process.

2. Forcibly stop the server process

   As instance manager, forcibly stop the server process using the pg_ctl command.

```
c:\>pg_ctl kill QUIT 896
```

8.11.3 Errors Caused by Failure to Stop PL/extJava

If an error occurs when PL/extJava is stopped and the instance fails to stop, take the following corrective action:

- Check if error messages have been output to the following logs.

  Refer to "Messages Output by PL/extJava" in Messages for information on the output error messages, and investigate the cause of the failure.

  - Event log
  - Enterprise Postgres server log
  - Container server log
  - Container Java VM log
  - Domain server log
  - Domain Java VM log

In some cases, processes may remain unduly, depending on the error that occurred when PL/extJava was stopped. Refer to "8.10.3 Errors Caused by Failure to Start PL/extJava" if this occurs, and end any processes that still exist.

If you cannot determine the cause of the error even after conducting the above investigation, use FJQSS to collect information, and then contact Fujitsu technical support.

8.12 Actions in Response to Error in a Distributed Transaction

If a system failure (such as server failure) occurs in an application that uses distributed transactions (such as .NET TransactionScope), then transactions may be changed to the in-doubt state. At that point, resources accessed by the transaction will be locked, and rendered unusable by other transactions.

The following describes how to check for in-doubt transactions, and how to resolve them.

How to check for in-doubt transactions

The following shows how to check for them:

If the server fails

1. An in-doubt transaction will have occurred if a message similar to the one below is output to the log when the server is restarted.

   Example

   ```
   LOG: Restoring prepared transaction 2103.
   ```

2. Refer to system view pg_prepared_xacts to obtain information about the prepared transaction.

   If the transaction identifier of the prepared transaction in the list (in the transaction column of pg_prepared_xacts) is the same as the identifier of the in-doubt transaction obtained from the log output when the server was restarted, then that row is the information about the in-doubt transaction.

   Example

   ```
   postgres=# select * from pg_prepared_xacts;
    transaction |   gid     |   prepared |  owner   | database
   -------------+-----------+------------+----------+----------
   2103 | 374cc221-f6dc-4b73-9d62-d4fec9b430cd | 2015-05-06 16:28:48.471+08 | postgres |
   postgres (1 row)
   ```

   Information about the in-doubt transaction is output to the row with the transaction ID 2103 in the transaction column.

If the client fails

If there are no clients connected and there is a prepared transaction in pg_prepared_xacts, then you can determine that the transaction is in the in-doubt state.

If at least one client is connected and there is a prepared transaction in pg_prepared_xacts, you cannot determine whether there is a transaction in the in-doubt state. In this case, use the following query to determine the in-doubt transaction from the acquired database name, user name, the time PREPARE TRANSACTION was executed, and the information about the table name accessed.

```
select gid,x.database,owner,prepared,l.relation::regclass as relation from pg_prepared_xacts x
left join pg_locks l on l.virtualtransaction = '-1/'||x.transaction and l.locktype='relation';
```

If it still cannot be determined from this information, wait a few moments and then check pg_prepared_xacts again.

If there is a transaction that has continued since the last time you checked, then it is likely that it is the one in the in-doubt state.

Point

As you can see from the explanations in this section, there is no one way to definitively determine in-doubt transactions.

Consider collecting other supplementary information (for example, logging on the client) or performing other operations (for example, allocating database users per job).

How to resolve in-doubt transactions

From the system view pg_prepared_xacts mentioned above, obtain the global transaction identifier (in the gid column of pg_prepared_xacts) for the in-doubt transaction, and issue either a ROLLBACK PREPARED statement or COMMIT PREPARED statement to resolve the in-doubt transaction.

Example

- Rolling back in-doubt transactions

```
postgres=# rollback prepared '374cc221-f6dc-4b73-9d62-d4fec9b430cd';
ROLLBACK PREPARED
```

- Committing in-doubt transactions

```
postgres=# commit prepared '374cc221-f6dc-4b73-9d62-d4fec9b430cd';
COMMIT PREPARED
```

8.13 I/O Errors Other than Disk Failure

Even if a disk is not defective, the same input-output error messages, as those generated when the disk is defective, may be output.

A few examples of such errors are given below. The appropriate action for each error is explained respectively.

- 8.13.1 Network Error with an External Disk
- 8.13.2 Errors Caused by Power Failure or Mounting Issues

8.13.1 Network Error with an External Disk

This is an error that occurs in the network path to/from an external disk.

Determine the cause of the error by checking the information in the event log and the server log, the disk access LED, network wiring, and network card status. Take appropriate action to remove the cause of the error, for example, replace problematic devices.

8.13.2 Errors Caused by Power Failure or Mounting Issues

These are errors that occur when the disk device is not turned on, automatic mounting of the disk was not set, or mounting was accidentally cancelled.

In this case, check the information in the event log and the server log, the disk access LED, and whether the disk is mounted correctly. If problems are detected, take appropriate action.

If mounting has been cancelled, it is possible that mounting was accidentally cancelled, or the existing setting (automatic mounting at the time of starting the operating system) has been changed so that mounting is not performed automatically. In this case, set the mounting to be performed automatically.

8.14 Operational Errors in PL/extJava Operations

If the pgx_jadmin command ends in an error, refer to the output error message, together with "Messages Output by PL/extJava" in Messages, and resolve the issue. If required, also refer to the PL/extJava log. Refer to "Appendix G PL/extJava Log Information" for information on the PL/extJava log.

The pgx_jadmin command runs in the PL/extJava environment, so any errors that might occur are almost always the same as those that occur when an instance is starting up in PL/extJava. Refer to "8.10.3 Errors Caused by Failure to Start PL/extJava" for information on typical examples.

8.15 Errors Related to Application Operations (during PL/extJava Operations)

8.15.1 Java Function Errors

If execution of a Java application fails, the Java function will return the error SQLSTATE 39000. Check the server log of Enterprise Postgres and then if required, check if any errors from IJServer12000 to IJServer12999 have been output to the logs below:

- Event log
- Container server log
- Container Java VM log
- Domain server log

- Web server trace log
- Web server internal log

Some examples are provided below.

Abnormal termination of the Java EE DAS service

Even if the Java EE DAS service ends in an error, it will be restarted automatically, enabling jobs to continue. However, if the error occurs repeatedly, investigate its cause by referring to "Messages Output by PL/extJava" in Messages, using the messages with "IJServer" in the message number that are output to the log file below. Identify the cause and prevent the error from recurring.

- Event log
- Domain server log

Abnormal termination of the PCMI service

The server instance(Java VM) monitored by the PCMI service and the Java EE DAS service will remain unduly. You must forcibly terminate these processes (refer to "8.10.3 Errors Caused by Failure to Start PL/extJava" for details), remove the cause of the error, and then restart the instance.

Java heap insufficiency

If an error message is output to the log below indicating that memory is insufficient and the Java application cannot be executed, then the heap area size and Perm area size of the server instance(Java VM) in the container must be modified.

- Container server log
- Container server instance(Java VM) log

Point

The following default values are set for the heap area and Perm area of the server instance(Java VM) in the container.

Type	Setting value
Server instance(Java VM) heap area size	512MB
Server instance(Java VM) Perm area size	192MB

Perform the following procedure to change the server instance(Java VM) heap area size and Perm area size in the container:

1. Use the pgx_jadmin command with the list-jvm-options subcommand to check the size of the heap area and Perm area in the current server instance(Java VM).

   #### Example

   ```
   > pgx_jadmin list-jvm-options --dbname db01
   ```

   Refer to the output results of the corresponding item to be checked below for the heap area size and the Perm area size.

Option	Item to be checked
heap area size	-Xmx
Perm area size	-XX:MaxPermSize

See

Refer to "pgx_jadmin" in the Reference for information on the list-jvm-options subcommand.

2. Stop the instance

   Stop the instance . Refer to 2.1 Starting and Stopping an Instance" for information on how to stop an instance.

   If the instance fails to stop, refer to "8.11 Actions in Response to Failure to Stop an Instance".

3. Use the pgx_jadmin command with the modify-jvm-options subcommand to modify the configured size of the heap area and Perm area in the server instance(Java VM).

   Example

   To change the heap area size and the Perm area size of the server instance(Java VM) to "1024 (megabytes)" and "384 (megabytes)", respectively.

```
> pgx_jadmin modify-jvm-options --dbname db01 --heapsize 1024 --permsize 384
```

   See

   Refer to "pgx_jadmin" in the Reference for information on the modify-jvm-options subcommand.

4. Start the instance

   Start the instance. Refer to 2.1 Starting and Stopping an Instance" for information on how to start an instance.

   If the instance fails to start, refer to "8.10 Actions in Response to Instance Startup Failure"

Domain root disk failure

Refer to "8.10.3 Errors Caused by Failure to Start PL/extJava" for information on the corrective action to take when the disk on which the domain root is placed fails, or when domain root resources become corrupted due to incorrect operation.

8.15.2 No Response from Java Functions

Possible causes for Java functions not responding for an extended period of time are described below. Take the corresponding corrective action accordingly.

A Java application is attempting to update a record that has already been updated by the source calling the Java function

The source calling the Java function and the Java application are handled as separate transactions. Therefore, if the Java application attempts to update a record that has already been updated by the Java function, the lock processing will never be processed. This kind of situation can be checked using the pg_locks view (refer to "Viewing Locks" in "Monitoring Database Activity" under "Server Administration" in the PostgreSQL Documentation for details).

The Java application processing includes a process that is taking a long time

Use the jstack command to collect a full thread dump for the process ID listed in the file below, and check the content of the process being executed (refer to "F.2 Notes on Using jstack" for information on the jstack command).

```
domainRoot\domains\domain1\config\pid
```

Appendix A Parameters

This appendix describes the parameters to be set in the postgresql.conf file of Enterprise Postgres.

The postgresql.conf file is located in the data storage destination.

- core_directory (string)

  This parameter specifies the directory where the corefile is to be output. If this parameter is omitted, the data storage destination is used by default. This parameter can only be set when specified on starting an instance. It cannot be changed dynamically, while an instance is active.

- core_contents (string)

  This parameter specifies the contents to be included in the corefile.

  - full: Outputs all contents of the server process memory to the corefile.
  - none: Does not output a corefile.
  - minimum: Outputs only non-shared memory server processes to the corefile. This reduces the size of the corefile. However, in some cases, this file may not contain sufficient information for examining the factor that caused the corefile to be output.

  If this parameter is omitted, "minimum" is used by default. This parameter can only be set when specified on starting an instance. It cannot be changed dynamically, while an instance is active.

- keystore_location (string)

  This parameter specifies the directory that stores the keystore file. Specify a different location from other database clusters. This parameter can only be set when specified on starting an instance. It cannot be changed dynamically, while an instance is active.

- tablespace_encryption_algorithm (string)

  This parameter specifies the encryption algorithm for tablespaces that will be created. Valid values are AES128, AES256, and none. If you specify "none", encryption is not performed. The default value is "none". To perform encryption, it is recommended that you specify AES256. Only superusers can change this setting.

- backup_destination (string)

  This parameter specifies the absolute path of the directory where pgx_dmpall will store the backup data. Specify a different location from other database clusters. This parameter can only be set when specified on starting an instance. It cannot be changed dynamically, while an instance is active.

  Place this directory on a different disk from the data directory to be backed up and the tablespace directory. Ensure that users do not store arbitrary files in this directory, because the contents of this directory are managed by the database system.

- search_path (string)

  When using the SUBSTR function compatible with Oracle databases, set "oracle" and "pg_catalog" in the search_path parameter. You must specify "oracle" before "pg_catalog".

Example

```
search_path = '"$user", public, oracle, pg_catalog'
```

Information

  - The search_path feature specifies the priority of the schema search path. The SUBSTR function in Oracle database is defined in the oracle schema.
  - Refer to "Statement Behavior" under "Server Administration" in the PostgreSQL Documentation for information on search_path.

- track_waits (string)

  This parameter enables collection of statistics for pgx_stat_lwlock and pgx_stat_latch.

  - on: Enables collection of statistics.
  - off: Disables collection of statistics.

  If this parameter is omitted, "on" is assumed.

  Only superusers can change this setting.

- track_sql (string)

  This parameter enables collection of statistics for pgx_stat_sql.

  - on: Enables collection of statistics.
  - off: Disables collection of statistics.

  If this parameter is omitted, "on" is assumed.

  Only superusers can change this setting.

Note

Note the following when specifying the path:

- Specify \\ as the path delimiter.
- Enclose the path in double quotes (") if it contains spaces.

See

Refer to "Server Configuration" under "Server Administration" in the PostgreSQL Documentation for information on other postgresql.conf parameters.

Appendix B System Administration Functions

This appendix describes the system administration functions of Enterprise Postgres.

See

Refer to "System Administration Functions" under "The SQL Language" in the PostgreSQL Documentation for information on other system administration functions.

B.1 WAL Mirroring Control Functions

The following table lists the functions that can be used for backup and recovery based on WAL mirroring.

Table B.1 WAL mirroring control functions

Name	Return type	Description
pgx_pause_wal_multiplexing()	void	Stops WAL multiplexing
pgx_resume_wal_multiplexing()	void	Resumes WAL multiplexing
pgx_is_wal_multiplexing_paused()	boolean	Returns true if WAL multiplexing has stopped

If WAL multiplexing has not been configured, these functions return an error. Setting the backup_destination parameter in postgresql.conf configures WAL multiplexing.

Only superusers can execute these functions.

B.2 Transparent Data Encryption Control Functions

The following table lists the functions that can be used for transparent data encryption.

Table B.2 Transparent data encryption control functions

Name	Return type	Description
pgx_open_keystore(*passphrase*)	void	Opens the keystore
pgx_set_master_key(*passphrase*)	void	Sets the master encryption key
pgx_set_keystore_passphrase(*oldPassphrase*, *newPassphrase*)	void	Changes the keystore passphrase

The pgx_open_keystore function uses the specified passphrase to open the keystore. When the keystore is opened, the master encryption key is loaded into the database server memory. In this way, you can access the encrypted data and create encrypted tablespaces. If the keystore is already open, this function returns an error.

Only superusers can execute this function. Also, this function cannot be executed within a transaction block.

The pgx_set_master_key function generates a master encryption key and stores it in the keystore. If the keystore does not exist, this function creates a keystore. If the keystore already exists, this function modifies the master encryption key. If the keystore has not been opened, this function opens it.

The passphrase is a string of 8 to 200 bytes.

Only superusers can execute this function. Also, this function cannot be executed within a transaction block. Processing is not affected by whether the keystore is open.

The pgx_set_keystore_passphrase function changes the keystore passphrase. Specify the current passphrase in *oldPassphrase*, and a new passphrase in *newPassphrase*.

The passphrase is a string of 8 to 200 bytes.

Only superusers can execute this function. Also, this function cannot be executed within a transaction block. Processing is not affected by whether the keystore is open.

Appendix C System Views

This appendix describes how to use the system views in Enterprise Postgres.

See

Refer to "System Views" under "Internals" in the PostgreSQL Documentation for information on other system views.

C.1 pgx_tablespaces

The pgx_tablespaces catalog provides information related to the encryption of tablespaces.

Name	Type	References	Description
spctablespace	oid	pg_tablespace.oid	Tablespace OID
spcencalgo	text		Tablespace encryption algorithm

The spcencalgo string displays one of the following values:

- none: Tablespace is not encrypted
- AES128: AES with key length of 128 bits
- AES256: AES with key length of 256 bits

C.2 pgx_stat_lwlock

The pgx_stat_lwlock view displays statistics related to lightweight locks, with each type of content displayed on a separate line.

Table C.1 pgx_stat_lwlock view

Column	Type	Description
lwlock_name	name	Name of the lightweight lock
total_waits	bigint	Number of waits caused by the lightweight lock
total_wait_time	double precision	Number of milliseconds spent in waits caused by the lightweight lock
stats_reset	timestamp with timezone	Last time at which this statistics was reset

C.3 pgx_stat_latch

The pgx_stat_latch view displays statistics related to latches, with each type of wait information within Enterprise Postgres displayed on a separate line.

Table C.2 pgx_stat_latch view

Column	Type	Description
latch_name	name	Name of the latch
total_waits	bigint	Number of waits caused a wait
total_wait_time	double precision	Number of milliseconds spent in waits caused by the latch
stats_reset	timestamp with timezone	Last time at which this statistic was reset

C.4 pgx_stat_walwriter

The pgx_stat_walwriter view display statistics related to WAL writing, in a single line.

Table C.3 pgx_stat_walwriter view

Column	Type	Description
dirty_writes	bigint	Number of times old WAL buffers were written to the disk because the WAL buffer was full when WAL records were added
writes	bigint	Number of WAL writes
write_blocks	bigint	Number of WAL write blocks
total_write_time	double precision	Number of milliseconds spent on WAL writing
stats_reset	timestamp with timezone	Last time at which this statistic was reset

C.5 pgx_stat_sql

The pgx_stat_sql view displays statistics related to SQL statement executions, with each type of SQL statement displayed on a separate line.

Table C.4 pgx_stat_sql view

Column	Type	Description
selects	bigint	Number of SELECT statements executed
inserts	bigint	Number of INSERT statements executed
deletes	bigint	Number of DELETE statements executed
updates	bigint	Number of UPDATE statements executed
declares	bigint	Number of DECLARE statements executed (number of cursor OPENs)
fetches	bigint	Number of FETCH statements executed
checkpoints	bigint	Number of CHECKPOINT statements executed
clusters	bigint	Number of CLUSTER statements executed
copies	bigint	Number of COPY statements executed
reindexes	bigint	Number of REINDEX statements executed
truncates	bigint	Number of TRUNCATE statements executed
locks	bigint	Number of times a lock occurred
stats_reset	timestamp with timezone	Last time at which this statistic was reset

Appendix D Activating and Stopping the Web Server Feature of WebAdmin

To use WebAdmin for creating and managing a Enterprise Postgres instance on a server where Enterprise Postgres is installed, you must first activate the Web server feature of WebAdmin.

This appendix describes how to activate and stop the Web server feature of WebAdmin.

D.1 Activating the Web Server Feature of WebAdmin

Follow the procedure below to activate the Web server feature of WebAdmin:

1. Display the [Services] window

   - Windows Server(R) 2012 or Windows Server(R) 2012 R2:

     In the [Start] screen, select [Administrative Tools], and then click [Services].

   - All other operating systems:

     In the [Start] menu, select [Administrative Tools], and then click [Services].

2. Start a service

   Select the displayed name "FUJITSU Enterprise Postgres WebAdmin *version*", and then click [Start Service].

You can also start a service by specifying the service name of the Web server feature of WebAdmin in the net start command or sc start command.

D.2 Stopping the Web Server Feature of WebAdmin

This section describes how to stop the Web server feature of WebAdmin.

Follow the procedure below to stop the Web server feature of WebAdmin:

1. Display the [Services] window

   - Windows Server(R) 2012 or Windows Server(R) 2012 R2:

     In the [Start] screen, select [Administrative Tools], and then click [Services].

   - All other operating systems:

     In the [Start] menu, select [Administrative Tools], and then click [Services].

2. Stop a service

   Select the displayed name "FUJITSU Enterprise Postgres WebAdmin *version*", and then click [Stop Service].

You can also stop a service by specifying the service name of the Web server feature of WebAdmin in the net stop command or sc stop command.

Appendix E Collecting Failure Investigation Data

If the cause of an error that occurs while building the environment or during operations is unclear, data must be collected for initial investigation.

This appendix describes how to collect data for initial investigation.

Use FJQSS (Information Collection Tool) to collect data for initial investigation.

See

Refer to the following manual for information on how to use FJQSS:

- Windows Server(R) 2012 and Windows Server(R) 2012 R2

  In the [Apps] menu, select [FJQSS (Information Collection Tool)], and then click [FJQSS User's Guide].

- Windows Server(R) 2008 R2 or earlier

  In the [Start] menu, select [FJQSS (Information Collection Tool)], and then click [FJQSS User's Guide].

Note

When using FJQSS to collect data for initial investigation, a window will be displayed for you to set the following environment variables:

- PGDATA

  Set the data storage destination.

- PGPORT

  Set the instance port number. This does not need to be set if the default port number (27500) has not been changed.

- PGUSER

  Set the database superuser.
  Set the database superuser so that client authentication is possible.
  FJQSS establishes a TCP/IP connection with the template1 database and collects data from the database.

In addition, when using database multiplexing, set the following environment variables:

- MCCONTROLDIR

  Refer to "Mirroring Controller Resources" in the Cluster Operation Guide for information on the Mirroring Controller management directory.

- The instance administrator user must perform FJQSS operations if using database multiplexing mode.

Appendix F Notes on PL/extJava

This section provides some points to note when using PL/extJava.

F.1 Thread Dump Tool

What is the thread dump tool?

The thread dump tool is a feature that outputs thread dumps, which are helpful in analyzing the execution status of programs as a whole or each thread, by outputting the status of each thread in the same manner as Java VM thread dumps (collected from the command prompt by pressing [Break] while holding down [Ctrl]).

The thread dump tool provides the following features in addition to Java VM thread dump output.

- File output
  Thread dumps can be output to files with the option (-f) specified.
- Header and footer output
  If a thread dump is output as standard output or standard error output, the thread dump will be included in the application log file (such as a file to which standard output of Servlet or EJB, or Java VM output is stored). The header and footer provide information that serves as a marker when referencing the log file. (Also output if the -f option is specified.)
  The header includes the thread dump collection datetime, command line, and memory usage status information.

Operating environment

JDK 7

Two types of commands

The thread dump tool provides the following two types of commands:

- The thdump command, used as a normal application (program that is started by the user)
- The thdumpSVC command, used as a Windows service (program that resides and operates inside the system)

Use these two commands according to the following:

- Use the thdump command if operating a Java program as a standard application.
- Use the thdumpSVC command if operating a Java program as a Windows service.

There are no functional differences between the two command types.

Storage destination directory

- thdump command::

```
Enterprise PostgresInstallDir\java\jdk7\tools\thdump
```

- thdumpSVC command:

```
Enterprise PostgresInstallDir \java\jdk7\tools\thdump\thdumpSVC
```

F.1.1 Using the thdump Command

Execute the thdump.exe command with the thdump.exe storage destination directory specified in the environment variable PATH, or with the full path specified.

Format

```
thdump.exe [-f logfile | -j] -p processid
```

or

```
thdump.exe [-f logfile | -j] program name with ".exe"
```

Refer to "F.1.3 Options" for information on each option.

How to use the thdump command

- If outputting thread dumps of applications started using "java.exe" as standard output of applications:

```
C:\> thdump java.exe
```

- If outputting a process thread dump for the process ID 123 to "a.log":

```
C:\> thdump -f a.log -p 123
```

- If outputting a thread dump of the applet that is operated using Internet Explorer with the JBK plugin to the JBK plugin error log (jbktrace.0):

```
C:\> thdump -j iexplore.exe
```

[jbkplugin.properties file]

```
jbk.plugin.debug.tracedir=C:\temp
jbk.plugin.debug.showvmmsg=true
```

F.1.2 Using the thdumpSVC Command

Use the thdumpSVC command to collect thread dumps of Java programs that are operated as a Windows service.
Use the following procedure to collect thread dumps:

1. Register a Windows service
2. Start the thread dump tool
3. Stop the thread dump tool
4. Delete the Windows service

You can collect multiple thread dumps by repeating steps 2 and 3.
When you have finished collecting thread dumps, perform step 4 "Delete the Windows service".

- Use "InstallUtil.exe" of Windows (R) internally to register and delete a Windows service.
- Use "sc.exe" to start and stop the thread dump tool.
- Thread dump tool operations must be performed from the command prompt by a user with administrator privileges. To run the thread dump tool from the command prompt as an administrator, right-click the command prompt executable file in Explorer, and then select "Run as administrator".
- To use the thdumpSVC command, you must install Microsoft(R) .NET FrameworkV2.0 in your operating environment.

Registering the thread dump tool as a Windows service

Format

```
thdumpSVC -Install
```

To register the thread dump tool as a Windows service, set the thdumpSVC command storage destination directory as the current directory, or specify the full path of the thdumpSVC command, and execute the command shown above.

Please note that only one thread dump tool can be registered on the same system. If you attempt to register two or more thread dump tools, an error will occur on execution of the command.

Checking the registration

You can use the Windows (R) administrative tool "Services" to check if the tool was successfully registered as a Windows service. If "Thdump Service" is displayed under "Name", this indicates that registration was successful.

If registration of the thread dump tool has already been completed, an error will occur, so use the Windows (R) administrative tool "Services" to check if the thread dump tool is registered.

Starting the thread dump tool

Format

```
sc start thdumpService [-f logfile | -j] -p processed
```

or

```
sc start thdumpService [-f logfile | -j] program name with ".exe"
```

Refer to "F.1.3 Options" for information on each option.

To start the thread dump tool, execute the command shown above. After the command is executed, thread dumps are collected.

Please note that only one thread dump tool can be registered on the same system. An error will occur if the command shown above is executed while the thread dump tool is running.

Checking if the tool has started

You can use the Windows (R) administrative tool "Services" to check if the thread dump tool was started successfully. If the "Status" is "Started", this indicates that the tool was started successfully.

After executing the command, messages from sc command control are displayed in the window.

If an error occurs during the start, one of the strings shown below is displayed. Messages output when an error occurred should be checked, but messages output at other times can be ignored as they will not cause issues.

- "[SC] StartService FAILED nnnn:" (nnnn is a number)
- "[SC] StartService: OpenService FAILED nnnn:" (nnnn is a number)

If the thread dump tool has not been registered, or if the thread dump tool has already started, an error will occur, so use the Windows (R) administrative tool "Services" to check the status of the thread dump tool.
Below are examples of messages that are output when an error occurs:

- If the thread dump tool has not been registered:
  "[SC] StartService: OpenService FAILED 1060:"
- If the thread dump tool has already started:
  "[SC] StartService FAILED 1056:"

Stopping the thread dump tool

Format

```
sc stop thdumpService
```

To stop the thread dump tool, execute the command shown above.

Checking if the tool has stopped

You can use the Windows (R) administrative tool "Services" to check if the thread dump tool was stopped successfully. If the "Status" is blank, this indicates that the thread dump tool was stopped successfully.

After executing the command, messages from sc command control are displayed in the window.

If an error occurs during the stop, one of the strings shown below is displayed. Messages output when an error occurred should be checked, but messages output at other times can be ignored as they will not cause issues.

- "[SC] ControlService FAILED nnnn:" (nnnn is a number)
- "[SC] OpenService FAILED nnnn:" (nnnn is a number)

If the thread dump tool has not been registered, or if the thread dump tool has already stopped, an error will occur, so use the Windows (R) administrative tool "Services" to check the status of the thread dump tool.

Below are examples of messages that are output when an error occurs:

- If the thread dump tool has not been registered:
  "[SC] OpenService FAILED 1060:"
- If the thread dump tool has already stopped:
  "[SC] ControlService FAILED 1062:"

Deleting the thread dump tool from Windows services

Format

```
thdumpSVC -Install -u
```

To delete the thread dump tool from Windows services, set the thdumpSVC command storage destination directory as the current directory, or specify the full path of the thdumpSVC command, and execute the command shown above.

Checking if the tool has been deleted

You can use the Windows (R) administrative tool "Services" to check if the tool was successfully deleted from Windows services. If "Thdump Service" is no longer displayed under "Name", this indicates that deletion was successful.

If deletion of the thread dump tool has already been completed (not registered), an error will occur, so use the Windows (R) administrative tool "Services" to check if the thread dump tool is registered.

Error log file

There is no console available for Windows services. Refer to the error log file to check if any errors occurred during collection of thread dumps.

Error log files are stored in the storage destination directory of the thdumpSVC command. Ensure that the storage destination directory of the thdumpSVC command is write-enabled.

Error log files are created during registration of Windows services, and are used to record errors that occur during the period up until deletion of Windows services. If thread dumps can be obtained without any issues, the error log file size will be 0 bytes.

F.1.3 Options

The options that can be specified when starting the thread dump tool are listed below.

Option	Description
-f logfile	Use this option to specify the thread dump output destination. If this option is omitted, thread dumps are output as standard output or standard error output. This cannot be specified at the same time as the -j option.
-j	Specify this option if using the Java VM output routine to output headers and footers. Specify this option for Java VM instances that are started with the vfprintf_hook option specified in the JNI_CreateJavaVM() function of JNI (Java Native Interface). Use this option if "jbk.plugin.debug.showvmmsg=true" is specified in the jbkplugin.properties file of the JBK plugin. This cannot be specified at the same time as the -f option.
-p processid	Specify the target process in this option using the process ID. This cannot be specified at the same time as the "program name with ".exe"" option below.
program name with ".exe"	Specify the target process in this option using a command name with the ".exe" suffix, such as "java.exe". If a command name is specified, the thread dump tool selects a process that was started using the specified command name, from amongst all of the processes that are currently running on the system, and handles it accordingly. If multiple processes were started from the same command, use the -p option to specify a process ID. This cannot be specified at the same time as the -p option.

Apart from the "program name with"exe"" option, you can specify the -f, -j, and -p options in any order.

Also, if the -f and -p options are specified multiple times, the last occurrence of the option will take effect.

When the command is executed, output is in the following order: header, footer, and stack trace.

The target process is specified using the process ID or program name with an ".exe" suffix (such as "java.exe"). You can use the Windows (R) "Task Manager" to check process IDs.

F.2 Notes on Using jstack

Note the following when using the jstack tool provided in JDK 7 for troubleshooting:

- You cannot use the -m option.

See

Information about the jstack tool is provided in JDK documentation. JDK documentation installed on this product is available from the following URL:

http://docs.oracle.com/javase/7/docs/

Appendix G PL/extJava Log Information

Errors and warnings in PL/extJava environments are output to log files.

This information is helpful for detecting whether an error has occurred, and if so, what was the cause.

G.1 Domain

This section explains the logs output by domains.

The logs described below are output by domains that control Java EE DAS services and containers.

G.1.1 Server Log

Output destination

The server log is output to the following file:

```
domainRoot\domains\domain1\logs\server.log
```

Output details

- Number of generations

  1

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the maximum log size If the maximum log size is exceeded, old information will be saved to a file with the original name followed by rotation datetime.

  [Example]

```
server.log_2012-01-07T15-23-30
```

Output example

```
 [#|2015-03-18T18:20:16.653+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1010: Entering Security
Startup Service|#]

[#|2015-03-18T18:20:16.653+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1143: Loading policy
provider com.sun.enterprise.security.provider.PolicyWrapper.|#]

[#|2015-03-18T18:20:17.230+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1115: Realm [admin-realm] of
classtype [com.sun.enterprise.security.auth.realm.file.FileRealm] successfully created.|#]

[#|2015-03-18T18:20:17.261+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1115: Realm [file] of
classtype [com.sun.enterprise.security.auth.realm.file.FileRealm] successfully created.|#]

[#|2015-03-18T18:20:17.277+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1115: Realm [certificate] of
classtype [com.sun.enterprise.security.auth.realm.certificate.CertificateRealm] successfully
created.|#]
```

G.1.2 Java VM Log

Output destination

The Java VM log is output to the following file:

```
domainRoot\pcmi\isje6\logs\server\console.log
```

Output details

- Number of generations

  1

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the maximum log size is exceeded, old information will be saved to a file with the original name followed by rotation datetime.

  [Example]

  ```
  console.log_2015_05_14-15_56_51
  ```

Output example

```
--------------------------------------------------------------------------------
                            02/09/2015  17:02:45
--------------------------------------------------------------------------------
C:\Program Files (x86)\Fujitsu\fsepserver32\java\jdk7\bin\java.exe
-XX:+PrintVMTerminatedMessage
-XX:InsufficientMemoryHandler=libpcmiJavaVM-win32-x86
-Xbootclasspath/a:C:\Program Files (x86)\Fujitsu\fsepserver32\java\jdk7\jre\lib\fmoni.jar
-Xrunfmoni:detail=y
-javaagent:C:\Program Files x86)\Fujitsu\fsepserver32\java\pcmi\lib\pcmi-java-premain.jar=C:\ryan
\domain\domain3\pcmi\isje6\work\process
\.PID_com.fujitsu.interstage.javaee.gf3adapter.Gf3PCMIAdapter_server
-cp
C:\Program Files (x86)\Fujitsu\fsepserver32\java\pcmi\lib\pcmi-api.jar;C:\Program Files x86)\Fujitsu
\fsepserver32\java\pcmi\lib\protocol.jar;C:/Program Files (x86)/Fujitsu/fsepserver32/java/fjje6/
glassfish/modules/glassfish.jar
-XX:PermSize=64m
-XX:MaxPermSize=192m
```

G.2 Container

This section explains the logs output by containers.

The logs described below are output by applications that run on the server instance (JavaVM).

G.2.1 Server Log

Output destination

The server log is output to the following file:

```
domainRoot\nodes\localhost-domain1\serverInstance(JavaVM)Name\logs\server.log
```

Output details

- Number of generations

  1

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the maximum log size If the maximum log size is exceeded, old information will be saved to a file with the original name followed by rotation datetime.

  [Example]

```
server.log_2012-01-07T15-23-30
```

Output example

```
[#|2015-03-18T18:20:16.653+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1010: Entering Security
Startup Service|#]

[#|2015-03-18T18:20:16.653+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1143: Loading policy
provider com.sun.enterprise.security.provider.PolicyWrapper.|#]

[#|2015-03-18T18:20:17.230+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1115: Realm [admin-realm] of
classtype [com.sun.enterprise.security.auth.realm.file.FileRealm] successfully created.|#]

[#|2015-03-18T18:20:17.261+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1115: Realm [file] of
classtype [com.sun.enterprise.security.auth.realm.file.FileRealm] successfully created.|#]

[#|2015-03-18T18:20:17.277+0900|INFO|||_ThreadID=1;_ThreadName=main;|SEC1115: Realm [certificate] of
classtype [com.sun.enterprise.security.auth.realm.certificate.CertificateRealm] successfully
created.|#]
```

Message details

The following explains the messages output in PL/extJava environments.

- Output format

  Messages are output in the following format:

```
[#|yyyy-MM-ddTHH:mm:ss.msTZD|logLevel|||_ThreadID=IDthreadId;_ThreadName=threadName;|
messageBody|#]
```

  - *logLevel*

    Fixed to "INFO".

  - *messageBody*

    Text output by the PL/extJava environment.

Note the following about the log content:

- If a user Java application detects an exception, only the class name of the exception is output.
- For security reasons, the exception stack trace and exception message are not output.

- Output format (message body)

  The message body is output in the following format:

```
logLevel: fsep-plextjava: [resultCode] message [backend_process=pid]
```

  - *logLevel*

    Fixed to " ERROR".

Log level	Output content
ERROR	Error information when an error occurs

  - *resultCode*

    The following return values are returned:

Return value	Explanation
0	Returned successfully.
1001	An internal error was detected. (protocol rule violation) Contact Fujitsu technical support.
2001	An internal error was detected. (protocol rule violation) Contact Fujitsu technical support.
2002	The Java application to be run cannot be found. Check if the jar file of the Java application has been registered, the FUNCTION specification contains an error, or the created Java application contains an error. [Considerations] - Was the "*packageName.className.methodName*" specified during registration of FUNCTION correct? - Does the "(Java function) arguments" specified during registration of FUNCTION match the type and number of arguments of the created Java application? - Does the return value type specified during registration of FUNCTION match the type of the return values for methods of the created Java application? - Have the methods of the created Java application been created as static methods (static) with the access modifier "public"?
3001	The user does not have privileges to run the Java application. Check the privileges for running Java applications. [Considerations] - Has the access modifier of the class of the created Java application been created as "public"?
3002	An exception occurred while the Java application was running. Check the database status or the processing content of the Java application.
9001	An internal error was detected. (environment error) Contact Fujitsu technical support.

  - backend_process

    PID of the Enterprise Postgres (backend) process.

G.2.2 Java VM Log

Output destination

The Java VM log is output to the following file: name of the server instance (Java VM)

```
domainRoot\pcmi\isje6\logs\serverInstance(JavaVM)Name\console.log
```

Output details

- Number of generations

  1

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the maximum log size If the maximum log size is exceeded, old information will be saved to a file with the original name followed by rotation datetime.

  [Example]

  ```
  console.log_2015_05_14-15_56_51
  ```

Output example

```
--------------------------------------------------------------------------------
                             05/09/2015  16:02:49
--------------------------------------------------------------------------------
C:\Program Files (x86)\Fujitsu\fsepserver32\java\jdk7\bin\java.exe
-XX:+PrintVMTerminatedMessage
-XX:InsufficientMemoryHandler=libpcmiJavaVM-win32-x86
-Xbootclasspath/a:C:\Program Files (x86)\Fujitsu\fsepserver32\java\jdk7\jre\lib\fmoni.jar
-Xrunfmoni:detail=y
-javaagent:C:\Program Files (x86)\Fujitsu\fsepserver32\java\pcmi\lib\pcmi-java-premain.jar=F:\fsep
\domain\pcmi\isje6\work\process\.PID_com.fujitsu.interstage.javaee.gf3adapter.Gf3PCMIAdapter_db01-1
-cp
C:\Program Files (x86)\Fujitsu\fsepserver32\java\pcmi\lib\pcmi-api.jar;C:\Program Files (x86)\Fujitsu
\fsepserver32\java\pcmi\lib\protocol.jar;C:/Program Files (x86)/Fujitsu/fsepserver32/java/fjje6/
glassfish/modules/glassfish.jar
-XX:PermSize=64m
-XX:MaxPermSize=192m
```

G.2.3 HTTP Access Log

Output destination

The HTTP access log is output to the following file:

```
domainRoot\nodes\localhost-domain1\serverInstance(JavaVM)Name\logs\access\server_access_log.txt
```

Output details

- Number of generations

  5

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  After one day, old information will be saved in a file that has the file name format "server_access_log.YYYY_MM_DD-hh_mm_ss.txt".
  If the rotation is performed within one second since the previous rotation, the file name format will be "server_access_log.YYYY_MM_DD-hh_mm_ss.*serialNumber*.txt".

Output example

```
"127.0.0.1" "-" "-" "14/May/2015:09:07:53 +0900" "POST /db01/callstored HTTP/1.1" "200" "1"
"127.0.0.1" "-" "4207" "ThreadID=68" "ThreadName=http-thread-pool-27532(1)" "127.0.0.1"
```

G.2.4 HTTP Trace Log

Output destination

The HTTP trace log is output to the following file:

```
domainRoot\nodes\localhost-domain1\serverInstance(JavaVM)Name\logs\http\trace.log
```

Output details

- Number of generations

  10

- Maximum log size

  10,485,760 bytes

- Length of Record

  Variable

- Rotation conditions

  If the maximum log size If the maximum log size is exceeded, old information will be saved to a file with the original name followed by rotation datetime, that is, "trace.log_YYYY_MM_DD-hh_mm_ss".
  If the rotation is performed within one second since the previous rotation, the file name format will be "trace.log_YYYY_MM_DD-hh_mm_ss.*serialNumber*".

Output example

```
"15/May/2015:11:21:31.608" "21(Grizzly-kernel-thread(1))" "conn" "127.0.0.1:60073"
"15/May/2015:11:21:31.609" "21(Grizzly-kernel-thread(1))" "qin" "127.0.0.1:60073"
"15/May/2015:11:21:31.609" "41(admin-thread-pool-27521(2))" "qout" "127.0.0.1:60073"
"15/May/2015:11:21:31.610" "41(admin-thread-pool-27521(2))" "recv" "GET /__asadmin/get?Xhelp=true
HTTP/1.1"
"15/May/2015:11:21:31.616" "41(admin-thread-pool-27521(2))" "send" "200" "127.0.0.1:60073"
"15/May/2015:11:21:31.626" "21(Grizzly-kernel-thread(1))" "k-wt" "127.0.0.1:60073"
"15/May/2015:11:21:31.704" "21(Grizzly-kernel-thread(1))" "qin" "127.0.0.1:60073"
"15/May/2015:11:21:31.704" "48(admin-thread-pool-27521(4))" "qout" "127.0.0.1:60073"
"15/May/2015:11:21:31.704" "48(admin-thread-pool-27521(4))" "recv" "GET /__asadmin/get?
DEFAULT=inst1.http-service.isjee-trace-log.max-history-files HTTP/1.1"
"15/May/2015:11:21:31.707" "48(admin-thread-pool-27521(4))" "send" "200" "127.0.0.1:60073"
"15/May/2015:11:21:31.711" "21(Grizzly-kernel-thread(1))" "k-wt" "127.0.0.1:60073"
"15/May/2015:11:21:31.758" "21(Grizzly-kernel-thread(1))" "qin" "127.0.0.1:60073"
"15/May/2015:11:21:31.758" "37(admin-thread-pool-27521(1))" "qout" "127.0.0.1:60073"
"15/May/2015:11:21:31.758" "21(Grizzly-kernel-thread(1))" "disc" "127.0.0.1:60073"
```

G.3 Web Server

This section explains the logs output by Web servers.

The logs described below are output by the web server that controls communication between the database and the server instance (JavaVM).

G.3.1 Error Log

Web server error information is output to an error log.

Output destination

The error log is output to the following file:

```
EnterprisePostgresInstallDir\java\ahs22\plextjava\domain1\logs\errorlog
```

Output details

- Number of generations

  5

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the log size is exceeded, old information will be saved to a file with the name "errorlog.N".

  N is a consecutive serial number, assigned in order of the newness of files - for the first file it is "0", and for the *n*th file it is "(N-1)".

- Output content

An error log is output every time an error occurs in a Web server.

The format of the log output as an error log is shown below.

```
[dateTime] [logLevel] [clientIpAddress] (errorNumber) errorDescription: messageBody
```

Output items

The following explains the output items in the output format.

- *dateTime*

  Datetime of error
  Output in the format "[*dayOfTheWeek month day hour*:*minute*:*second year*]".

- *logLevel*

Log level	Error log multiplicity	Output at initial setup
emerg	Error generated during an emergency	Y
alert	Error that prevents operations if not corrected	Y
crit	Error that should be handled immediately	Y
error	Minor error that can be ignored	Y
warn	Warning that can be ignored	Y
notice	Notification that can occur at any time and should be noted down	Y

Log level	Error log multiplicity	Output at initial setup
info	Notification other than notice Output when an environment definition is edited.	N
debug	Log during module development and debugging Output when an environment definition is edited.	N

- *clientIpAddress*

  IP address of the server, such as a client or proxy server
  Output when an error occurs during Web server access from a client.
  This item may be omitted.

- *errorNumber*

  Operating system error number
  This item may be omitted.

- *errorDescription*

  Description of the error number
  This item may be omitted.

- *messageBody*

  Body of the message

Output example

```
[Fri Sep 05 16:03:10 2015] [notice] Child 10752: Child process is running
[Fri Sep 05 16:03:10 2015] [notice] Child 10752: Acquired the start mutex.
[Fri Sep 05 16:03:10 2015] [notice] Child 10752: Starting 3000 worker threads.
```

G.3.2 Trace Log

Web server input/output information is output to a trace log.

Output destination

The trace log is output to the following file:

```
EnterprisePostgresInstallDir\java/ahs22\plextjava\domain1\logs\tracelog
```

Output details

- Number of generations

  5

- Maximum log size

  2,097,152 bytes

- Length of Record

  Variable

- Rotation conditions

  If the log size is exceeded, old information will be saved to a file with the name "tracelog.N".

  N is a consecutive serial number, assigned in order of the newness of files - for the first file it is "0", and for the *n*th file it is "(N-1)".

- Output content

  The trace log is output at the times shown below for each item of trace information.

- Web client input/output information
  - When a TCP connection is established
  - When a TCP connection is closed
  - When an HTTP request is received
  - When an HTTP response is sent
- Plug-in module input/output information
  - When the plug-in module response processing function is called
  - When the plug-in module response processing function is returned

Output format

The format of the log output as a trace log is shown below.

```
[dateTime][processId][threadId]detailedEventInformation
```

Output items

The following explains the output items in the output format.

- *dateTime*

  Datetime of trace information output
  The datetime is output in the format "[*day*/*month*/*year*:*hour*:*minute*:*second*.*millisecond*]".

- *processId*

  Process ID of the daemon process

- *threadId*

  Thread ID of the communication thread

- *detailedEventInformation*

  The following table explains the trace content formats for each output trigger.

Output trigger	Event	Detailed event information
When a TCP connection is established	conn	*IpAddrOfClient*:*portNum* =>*IpAddrOfTargetWebServer*:*PortNum* I
When a TCP connection is closed	disc	
When an HTTP request is received	recv	Content of the request line This portion contains the escaped request.
When an HTTP response is sent	send	Status code
When the plug-in module response processing function is called (*1)	call	Module source name (*2)
When the plug-in module response processing function is returned (*1)	rtn	Module source name (return code) (*2) The following are output as return codes: - When response processing in this module is not executed: -1 - When response processing in this module is executed: Other than -1

*1: The output target plug-in modules are those not provided in HTTP Server 2.2.

*2: This is the source file name when the plug-in module is compiled. The module source name of the Web server connector provided in the Servlet service is output as "mod_jk2.c".

Information

The HTTP Keep-Alive feature connection retention time can be checked from the trace log. For the output trace log, the datetime of disc (event when a TCP connection is closed) and send (event when an HTTP response is sent) is checked, and calculated according to the following formula:

```
httpKeepAliveConnectionRetentionTime= discDatetime - sendDatetime
```

Note: When a timeout occurs for a request from the web server to break the TCP connection to the client, if the disconnection notice from the client cannot be received within two seconds, the connection retention time may be two seconds longer than the setting in the KeepAliveTimeout directive..

Output example

```
[15/May/2015:11:34:34.550][5144][5157]conn 127.0.0.1:44217=>127.0.0.1:27530
[15/May/2015:11:34:34.550][5144][5157]recv "POST /db01/callstored HTTP/1.1"
[15/May/2015:11:34:34.551][5144][5157]call mod_jk2.c
[15/May/2015:11:34:34.576][5144][5157]rtn  mod_jk2.c(0)
[15/May/2015:11:34:34.576][5144][5157]send 400
[15/May/2015:11:34:34.591][5144][5157]disc
```

G.3.3 Access Log

In HTTP Server 2.2, the access status from the Web browser is output to an access log.

Output destination

The access log is output to the following file:

```
EnterprisePostgresInstallDir\java/ahs22\plextjava\domain1\logs\accesslog
```

Output details

- Number of generations

  5

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the log size is exceeded, old information will be saved to a file with the name "accesslog.N".

  N is a serial number, assigned consecutively starting at 0. The serial numbers are assigned in order of the newness of files, with the first file having the serial number "0" and the *n*th file having the serial number "(N-1)".

- Output content

  The access log is output when the Web server receives a request from a client and sends a response to the client.

Output format

The format of logs output in the default value format for the access log (ahs-analysis) is shown below. Items not specified during access are output as a hyphen "-".

```
hostName userNameIdentifier userName dateTime "request" statusCode dataTransferVolume
ipAddressOfWebServer:portNumber hostHeader processId processingTime requestId
```

Output items

The following table explains the output items in the output format.

Output item	Output content
hostName	IP address or host name of the client .
userNameIdentifier	Personal user information returned from the client.
userName	User name sent from the client.
dateTime	Datetime at which the request was received from the client. Output in the format "[*day*/*month*/*year*:*hour*:*minute*:*second timeDifferenceFromGMT*]".
request	Request content from the client.
statusCode	Code returned to the client.
dataTransferVolume	Amount of data transferred to the client.
ipAddressOfWebServer:*portNumber*	IP address and port number of the Web server that received the request.
hostHeader	Host header content sent from the client.
processId	Process ID of the process that processed the request.
processingTime	Time from when the request was received until processing was completed. Output in the format "*microsecond*".
requestId	Unique ID granted per request

Output example

```
127.0.0.1 - - [15/May/2015:11:32:17 +0900] "POST /db01/callstored HTTP/1.1" 200 3 127.0.0.1:80
127.0.0.1 5143 26901 -
```

G.3.4 Internal Log

Web server connector error information is output to a log.

Output destination

Error logs are output to the following file:

```
EnterprisePostgresInstallDir\java\wsc\logs\jk2\httpd_domain1.conf\jk2.log
```

Output details

- Number of generations

  2

- Maximum log size

  1,048,576 bytes

- Length of Record

  Variable

- Rotation conditions

  If the specified log size or the specified time is exceeded, old information will be saved to a file with the name "jk2_YY.MM.DD_hh.mm.ss.log".

Output example

```
    [05/09/2015 16:03:10:382 +0900] ( info) IJServer12047: Web Server Connector running. conf="C:
\Program Files (x86)\Fujitsu\fsepserver32\java\wsc\conf\jk2\httpd_domain1.conf\workers2.properties"
pid=10752 tid=16400
[05/09/2015 16:04:37:285 +0900] ( info) IJServer12048: Web Server Connector stops running. conf="C:
\Program Files (x86)\Fujitsu\fsepserver32\java\wsc\conf\jk2\httpd_domain1.conf\workers2.properties"
pid=10752 tid=16400
```

Index